이학석사 학위논문

$UsnJrnl 파일을 이용한 사용자 행위 추적 연구

A study on user behavior tracking using $UsnJrnl

2018년 2월

서울대학교 융합과학기술대학원

수리정보과학과 디지털포렌식 전공

윤 효 진

$UsnJrnl 파일을 이용한 사용자 행위 추적 연구

A study on user behavior tracking using $UsnJrnl

지도교수 천 정 희

이 논문을 이학석사 학위논문으로 제출함

2018년 2월

서울대학교 융합과학기술대학원

수리정보과학과 디지털포렌식 전공

윤 효 진

윤효진의 석사 학위논문을 인준함

2018년 2월

위 원 장 이 광 근 (인)

부 위 원 장 천 정 희 (인)

위 원 박 상 준 (인)

국문초록

디지털포렌식 수사관이라면 누구나 범죄의 행위를 규명하기 위해 많은 고민을 하게 된다. 사건의 유형이나 사용자의 행위 패턴, 혹은 사건의 쟁점에 따라 압수의 대상이 달라지고 압수물에 대한 분석 방향이 결정된다. 예컨대, 기술 유출 사건의 경우라면 혐의와 관련된 문건이 특정 기기에 존재 했었는지에 대한 여부, 혹은 기술이 유출된 경로를 확인하는 것으로 분석이 시작되고, 증거 인멸의 사건이라면 혐의와 관련된 문건이 무엇인지, 언제, 어떤 방법으로 삭제가 되었는지, 혹은 삭제된 문건을 어떻게 찾아내고 복구할지에 대한 파악으로 시작된다.

디지털 기술의 보편화, 대중화로 인해 개인의 일상과 밀접한 관계를 맺게 되면서 증거로서의 디지털 정보가 가지는 의미는 점차 증대되고 있다. 반면 이러한 기술이 고도화 되고 새로운 기술 지식에 대한 접근이 용이해짐에 따라 개인들의 디지털 정보에 대한 지식이 지능화 되어 수사기관의 개개인의 디지털 정보에 대한 접근은 어려워지고 있다. 특히, 개인의 프라이버시나 기업의 보안 의식이 강화되면서 안티포렌식 기술 또한 더욱 다양하고 정밀해 짐에 따라 이러한 디지털 증거를 분석하고 그 결과를 현출해야 하는 디지털포렌식 수사관들에게는 더 다양하고 심도있는 기술이나 정보 습득이 요구되어진다.

디지털 증거 분석 요청의 대부분은 사용자의 특정 행위에 대한 시간 정보다. 특히 쟁점이 되는 행위의 실행 시간, 예를 들어 특정 파일이 존재한다면 그 파일을 생성하거나 변경, 삭제한 시간 등에 대한 정보다. 모든 사건에 있어서 시간정보는 범죄 행위를 규명하는 필수적 요소이고, 디지털 증거에 있어서도 예외는 아닐 것이다. 디지털포렌식의 관점에서 이러한 정보들을 확인할 수 있는 요소들은 많지만 그 중 본 연구에서 다루었던 $UsnJrnl[1] 파일은 그 기능의 특성상 사용자 및 시스템의 행위를

1) NTFS 파일 시스템에서 파일, 디렉토리 및 기타 NTFS 파일 시스템 객체가 추가, 삭제, 수정되면 해당 변경 사항을 기록하는 파일이다. Microsoft, ‘Change Journal’ https://msdn.microsoft.com/en-us/library/windows/desktop/aa363798(v=vs.85).aspx, 2017.12.2.

비교적 구체적인 시간정보와 함께 기록하고 있다. 하지만 현재 국·내외적으로 디지털포렌식 수사관들이 활용할 수 있는 $UsnJrnl 파일 관련 연구는 부족한 실정이다. 다양한 수사기법에 대한 연구는 실무자들에게 사건을 접근하는 방법을 제시하고 사건을 해결하는데 유용한 자료가 되는 바, 이번 연구를 통해 시간정보를 기반으로 하는 사용자의 행위 추적에 용이한 $UsnJrnl 파일을 활용한 분석 방법을 제안하고자 했다.

본 연구에서는 실제 실무에서 $UsnJrnl 파일로부터 탐지된 정보들을 활용한 사례를 보여주었고, 가상의 시나리오를 구성하여 구체적인 분석 방법을 제시했다. 해당 사례 분석을 통해 사건의 쟁점이 되는 파일의 삭제 시간 정보를 확인할 수 있었는데, 특히 완전 삭제 프로그램을 사용하거나 기타 안티포렌식 행위들에 대해서도 그 흔적을 추적해 나갈 수 있음을 보여주었다.

$UsnJrnl 파일이 남기는 무수히 많은 정보들 중에서 필요한 정보만을 탐색하는 것이 쉬운 것만은 아니었다. 그러나 단 1퍼센트의 가능성이라도 있다면 쉽게 간과할 수는 없을 것이다. 이번 연구를 통해 더 다양하고 활발한 연구의 필요성을 느끼면서 동시에 본 연구가 우리 디지털포렌식 수사관들에게 유용하게 활용되기를 바란다.

주요어 : $UsnJrnl, 행위 추적, 시간 정보, 디지털포렌식, 안티포렌식
학 번 : 2016-26057

목 차

표 목 차

그 림 목 차

제 1 장　연구의 목적과 방법

모든 사건에서 사용자의 특정 행위에 대한 시간 정보는 중요한 의미를 갖는다. 특히 범죄를 특정함에 있어 단 몇 시간, 혹은 몇 분이라는 짧은 시간은 그 혐의를 규명하는데 많은 영향을 끼친다. 행위자가 언제 어느 시점에 어떤 행위를 했는지는 사건을 해결하는데 있어 반드시 파악해야 할 중요한 요소이며 특히 쟁점이 되는 것이 문서라면 그 문서가 언제 생성되었고, 언제 수정되었으며, 언제 삭제되었는가는 중요한 단서가 되고 결정적인 증거가 되기도 한다. 실무에 있어서도 포렌식 분석 요청의 상당수는 사용자의 특정 행위에 대한 시간 정보이다. 따라서 이러한 경우 사용자의 일련의 행위 등을 고려한 후 이에 대한 객관성을 확보하기 위해 철저한 테스트를 거쳐 사용자 혹은 시스템의 행위, 실행 시간을 추정하거나 특정 짓는다.

최근 서울중앙지검 방위사업수사부에서는 방산비리 혐의로 KAI 한국항공우주산업주식회사와 그 관련업체를 대상으로 대대적인 압수수색을 진행했다. 압수수색 당시 KAI 사의 직원 대다수가 영구 삭제 프로그램인 'Eraser'를 사용하고 있었다. 'Eraser' 프로그램은 삭제된 파일의 복구를 불가능하게 만드는 강력한 안티포렌식 프로그램이다. 당시 이 프로그램 사용에 대해 검찰은 적극적인 증거 인멸을 시도한 것이라고 주장했고 피압수측은 보안 강화를 위한 국방부 훈령에 따른 내부 정책이라고 주장하면서 두 의견이 팽팽히 맞섰다[2]. 만약 이 사건에서 그들이 삭제한 파일이 언제 삭제되었고, 또 삭제된 파일이 어떤 파일이었는지, 혹은 문제가 되었던 행위가 특정 시기에만 일어났는지, 시간적으로 일관성 있게 혹은 설득력 있게 이루어 졌는지에 대한 확인이 가능하다면 이러한 쟁점은 쉽게 정리가 될 수도 있다.

개인 프라이버시는 물론 기관, 기업의 보안 의식이 강화되면서 안티포렌식은 디지털 정보를 이용하는 사람들에게 점점 일반화, 일상화 되어가

2) 인터넷 기사, '檢, KAI 증거인멸 정황포착…" 삭제전용 프로그램 최근 대거 가동"(종합)' 연합뉴스, 2017.7.19.

고 있다. 범죄와의 관련성을 떠나 디지털 데이터 포화 상태의 개인들에게 안티포렌식이 일상이 되는 것은 당연한 일일 것이다. 개인프라이버시의 강화, 안티포렌식을 포함한 디지털 기술의 발전은 수사기관의 디지털 포렌식 분석 환경을 더욱 어렵게 만드는 것은 사실이지만 그럴수록 수사기관은 다양한 방법으로 접근을 시도해 봐야 한다.

이번 연구는 이러한 안티포렌식의 흔적을 발견했을 때 과연 수사기관에서 어떠한 방법으로 접근해야 하는지에 대한 고민으로 시작되었다. 안티포렌식의 행위가 과연 범죄 사실의 고의를 가졌는지에 대한 판단은 과거에서부터의 일련의 행위를 통해 파악하거나, 그 행위가 발생한 구체적인 시간 정보 파악이 가능하다면 사건 해결의 실마리가 될 수도 있다.

이러한 정보를 수집할 수 있는 경로는 여러 가지가 있는데 본 연구에서는 NTFS 파일 시스템의 저널파일 중 $UsnJrnl 파일에 중점을 두고 연구를 진행했다. 저널파일이란 시스템을 운용하면서 장애가 발생하는 경우 복구를 위해 데이터를 저장하거나 관리하는 파일로 시스템의 전반적인 행위를 기록한다. 윈도우 운영체제의 저널파일로는 $LogFile과 $UsnJrnl 파일이 있는데, 특히 $UsnJrnl 파일은 다른 파일들과 달리 파일 레코드 자체에 독립적인 시간 정보를 포함하고 있어 시간의 흐름에 따라 사용자의 행위를 파악할 수 있는 유용한 파일이다. 하지만 실제 분석 실무에서는 위 파일을 활용한 사례는 드물다. 이는 데이터가 잔존하는 기간이 짧고 여러 시스템 영역의 기록들이 혼재되어 있어 사용자 영역에 대한 구분이 다소 어렵기 때문일 것이라 판단된다. 하지만 대부분의 디지털 정보가 그러하듯 사용자의 습성에 따라 혹은 시스템의 환경에 따라 큰 편차가 존재한다는 한계점은 가지고 있으나 만약 사건과 연결지을 수 있는 데이터를 발견하게 된다면 이 $UsnJrnl 파일의 정보는 그 어떤 다른 데이터들 보다 강력하고 구체적인 단서가 될 수 있을 것이다.

본 논문은 총 7장으로 구성되었다. 제 1장에서는 본 논문의 주요 목적과 주된 기술 방법을 기술하고, 제 2장에서는 본 논문의 주요 연구 대상인 $UsnJrnl 파일에 대한 기존의 연구들을 알아볼 것이며, 제 3장에서 $UsnJrnl 파일의 기능, 구조 등 기본 개념에 대해 구체적으로 기술함과

동시에 이 파일과 밀접한 관련이 있는 $LogFile 및 MFT Entry에 대해서도 함께 비교하여 볼 것이다. 또한 이 $UsnJrnl 파일이 MFT Entry와 어떻게 연결되는지, 삭제된 MFT Entry 정보를 어떻게 확인할 수 있는지에 대해 기술하였다. 이어서 제 4장에서는 위 $UsnJrnl파일이 가지는 증거로서의 가치 및 의미를 논하여 보고 이 파일 수집의 필요성과 근거, 그 방법에 대해 알아본다. 제 5장에서는 실제 검찰청의 디지털 증거 분석 사례를 통해 $UsnJrnl 파일을 어떻게 활용해 왔는지에 대해 살펴보고 제 6장에서는 증거인멸 사례와 기술유출 사례 두 가지의 가상의 시나리오를 통해 구체적인 분석 방법과 활용 방안에 대해 논하고자 한다. 마지막 제 7장에서 본 연구에 대한 최종 결과와 향후 연구 방향에 대해 기술하겠다.

제 2 장 관련 연구

1. InPrivate 모드[3] 및 CCleaner[4] 사용 흔적 연구[5]

Christopher John Lees는 Inprivate 모드와 CCleaner 프로그램 사용 테

---

3) 마이크로소프트사에서 제공하는 인터넷 익스플로러 기능 중의 하나로, 일반적으로는 브라우저가 웹의 환경을 향상시키기 위해 검색 기록과 같은 일부 정보를 저장하는데 Inprivate 브라우징을 사용하는 경우 탭을 닫으면 암호, 검색 기록 및 페이지 기록과 같은 정보가 삭제된다.
Microsoft, 'Internet Explorer 11의 보안 및 개인 정보 설정 변경' http://support.microsoft.com/ko-kr/help/17479/windows-internet-explorer-11-change-security-privacy-settings, 2017.11.30.

4) 잠재적으로 불필요한 파일과 잘못된 윈도우 레지스트리 항목을 제거하는 피리폼사의 유틸리티이다.
위키백과, 'CCleaner' http://ko.wikipedia.org/wiki/CCleaner, 2017.11.24.

5) Christopher Lees, "Determining removal of forensic artefacts using the USN change journal" Digital Investigation vol.10, no.4, 300-310(2013)

스트에서 $UsnJrnl 파일이 수사의 단서로 활용될 수 있음을 연구한 바 있다.

본 연구에서는 $UsnJrnl 파일을 분석하여 InPrivate 모드를 실행하고 종료하는 동안 시스템이 실행시키는 프로세스와 접근 경로 등에 대해 구체적으로 확인해 주었다. 더불어 파일 삭제 프로그램인 CCleaner 프로그램 사용시 실행되는 프리패치 파일 정보, 파일 삭제시의 변화, CCleaner 설정 변경에 따른 변화 등을 확인해 주었다.

위 연구 결과에 따르면 InPrivate 모드 종료시 해당 모드에서 접속했던 인터넷 접근 이력이 일괄 삭제되는데, 이때 삭제되는 파일들 대부분이 Temporary Internet files 폴더에 존재하는 파일들임을 확인할 수 있다.

또한 CCleaner 실험에서도 $UsnJrnl 파일에서 해당 프로그램의 사용 흔적을 확인할 수 있었는데, 설정을 달리하여 테스트함으로써 각 설정별로 어떻게 기록되는지도 함께 확인해 주었다. 구체적으로는 덮어쓰지 않는 경우, 한 번 덮어쓰는 경우, 세 번 덮어쓰는 경우로 나누어, 덮어쓰지 않는 경우 '파일 삭제, 속성 변경'의 이벤트가 발생하고, 한 번 덮어쓰는 경우 '속성 변경, 데이터 덮어쓰기, 파일명 변경, 파일 삭제'의 이벤트가 실행된다. 세 번 덮어쓰는 경우도 위 한 번 덮어쓰기를 실행할 때와 동일함을 확인하였다.

2. $UsnJrnl 및 $LogFile의 조합을 통한 MFT Entry 분석 연구[6)]

Frank Uijtewaal과 Jeroen van Prooijen은 그들의 연구에서 $UsnJrnl 파일이 $LogFile과의 조합을 통해 덮어씌워진 MFT Entry의 덮어씌워지기 전 정보를 확인할 수 있음을 확인하였다. 그들은 실험을 통해 파일을 결합하기 전과 후를 비교하여 결과를 보여주고 있는데, 두 저널 파일이 가지고 있는 레코드 정보를 추적해 이미 덮어쓰여진 MFT Entry에 대해서도 분석이 가능하다는 것을 증명하였다.

---

6) Frank Uijtewaal, Jeroen van Prooijen, "UsnJrnl Parsing for File System History Project Report" University of Amsterdam, work4.delaat.net, (2016)

삭제된 MFT Entry라도 다른 파일에 의해 덮어쓰이지 않았다면 복구가 가능하다. 하지만 이미 덮어쓰여진 경우라면 그 정보를 확인하기 어렵다. 하지만 동일한 MFT Entry 번호를 사용했던 $UsnJrnl 파일과 $LogFile이 가지는 정보를 통해 해당 레코드로부터 MFT Entry 정보를 확인할 수 있음을 확인하였다.

본 연구에서는 상태가 다른 네 개의 파일을 서로 비교하였다. 사용 중인 파일과 삭제된 파일, 덮어쓰여지기 전후 파일이다. 이 중 덮어쓰여지기 전후 파일은 동일한 MFT Entry를 사용한다. 네 개의 파일 중 덮어쓰여진 파일은 복구가 불가능하다. 그러나 MFT Entry 번호를 사용하여 $UsnJrnl 파일과 $LogFile을 분석하면 덮어쓰여지기 전 파일에 대한 정보를 확인할 수 있음을 보여주었다.

제 3 장 NTFS 파일 시스템의 저널파일

1. $UsnJrnl 파일 및 관련 파일 개요

본 장에서는 이번 연구의 주요 파일인 $UsnJrnl 파일에 대해서 구체적으로 설명하고 위 파일과 연관되어 있는 MFT Entry와 $LogFile에 대해서는 비교적 간략하게 알아보고자 한다. 기술 순서는 세 파일의 기준이 되는 MFT Entry에 대해 우선 설명하고 이후에 MFT Entry와의 연관성을 기준으로 $UsnJrnl과 $LogFile에 대해 설명하도록 하겠다.

각 파일들에 대한 개요 이후에는 세 파일들을 연결하는 방법과 $UsnJrnl 파일을 통해 삭제된 MFT Entry 정보를 알 수 있는 원리에 대해 중점적으로 기술하고자 한다.

(1) MFT Entry

NTFS[7](New Technology File System)는 Windows NT 계열 운영체제의 파일 시스템으로 이 파일 시스템은 모든 파일과 디렉토리를 MFT(Master File Table)에서 관리한다. 이 MFT는 개별 파일들의 정보를 저장하는 레코드인 MFT Entry 들로 구성되어 있고 MFT Entry의 크기는 $Boot에서 정의되나 기본적으로는 1,024 바이트의 크기를 가진다. 시스템 내 모든 파일이나 디렉토리는 하나 이상의 MFT Entry를 가지는데 이 MFT Entry에는 해당 파일의 위치, 시간 정보, 파일 이름, 크기 등의 메타데이터 정보를 저장하고 있다.[8]

MFT Entry는 헤더 영역과 속성 영역으로 구분된다. 헤더 영역에는 메타데이터가 업데이트 될 때 생성되는 로그 레코드 번호인 LSN ($LogFile Sequence Number)과 MFT Entry의 사용 횟수를 알 수 있는 Sequence Value를 포함한다.

MFT Entry는 파일 타입에 따라 다른 속성을 가지고 있으나 그 중 $STANDARD_INFORMATION 속성과 $FILE_NAME 속성은 대부분의 MFT Entry가 가지고 있는 공통되는 속성이다. $STANDARD_INFORMATION 속성에는 파일의 시간정보, 거주, 비거주 여부, 마지막 USN(Update Sequence Number) 정보([그림3-11]참조)를 기록하고 $FILE_NAME 에는 파일 이름(Unicode)과 시간 정보, 부모 MFT 참조 주소 정보 등을 기록한다.

모든 MFT Entry에는 $DATA 속성을 가지고 있으며 그 데이터의 양이 약 700 바이트 정도인 경우 MFT Entry 내에 데이터를 기록(거주 속성)하고, 그 이상인 경우 그 데이터의 위치를 알려주는 런리스트의 값을 기록한다.

---

7) 위키백과, 'NTFS' https://ko.wikipedia.org/wiki/NTFS, 2017.12.2.
Microsoft, 'NTFS 개요' https://msdn.microsoft.com/ko-kr/library/dn466522(v=ws.11).aspx, 2017.12.2.

8) Brian Carrier, 「File System Forensic Analysis」 Addison Wesley(2006), 274-275

0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
Signature				Offset to Fixup Array		Fixup array Entry Number		LSN ($LogFile Sequence Number)							
Sequence Value		Link Count		Offset to First Attr		Flags		Used Size of MFT Entry				Allocated Size of MFT Entry			
File Reference to Base MFT Entry								Next Attr ID							
Fixup Array								Attribute type ID				Length of Attribute			
Reg Flag	Name Length	Offset to Name		Flags		Attribute ID		Size of Content				Offset of Content		INDX Flag	Unused
Creation Time								Modified Time							
MFT Modified Time								Accessed Time							
Flags				Max No. of Ver				Ver. No.				Class ID			
Owner ID				Security ID				Quata Charged							
USN (Update Sequence Number)								Attr type ID				Length of Attr			
Reg Flag	Name Length	Offset to Name		Flags		Attr ID		Size of Content				Offset of Content		Unused	
File Reference of Parent directory								Creation Time							
Modified Time								MFT Modified Time							
Accessed Time								Allocated Size							
Using Allocation Size								Attr Flag				Reparse value			
Name Length	Length size	File Name													
Attr type ID				Length of Attr				Reg Flag	Name Length	Offset to Name		Flags		Attr ID	
General Header															
Size of content				Offset to content											

[그림3-1] $STANDARD_INFORMATION,$FILE_NAME,$DATA(Resident) 속성을 갖는 MFT Entry 구조

(2) $UsnJrnl

Windows 2000에 NTFS 5.0 버전부터 $UsnJrnl을 포함했다. 이 $UsnJrnl은 메타데이터를 구성하는 파일로 $UsnJrnl 이라는 이름을 가

진다.[9] 디렉토리, 파일 등 NTFS 파일 시스템이 관리하는 파일들이 변경되는 경우 마지막 전체 백업 이후의 변경 사항만을 기록하는 일종의 로그 파일로 시스템에서 자동백업 시스템을 사용하는 경우 특정 시점에 모든 프로그램 정보나 데이터 정보를 백업하는 것은 효율성이나 경제성의 측면에서 비효율적이기 때문에 변경된 사항만을 기록한다. 볼륨의 변경을 확인하기 위한 용도로만 사용되므로 비정상적으로 종료된 작업에 대해 롤백을 수행하지는 못한다.

이 파일은 Windows 7 부터 운영체제가 위치한 드라이브에 기본으로 활성화 되어 있고, $UsnJrnl 파일이 활성화되어 있는 해당 볼륨에 대한 변경 사항을 기록하며 아래의 명령어를 통해 다른 드라이브에서 선택적으로 활성화가 가능하다.

fsutil usn <createjournal> m=<maxsize> a=<allocationdelta> <volumepath>[10]

① $UsnJrnl의 MFT Entry 구조와 $DATA 속성 정보

$UsnJrnl 파일은 $Extend 디렉토리 하위에 존재하고 그 구조는 MFT Entry 헤더와 $STANDARD_INFORMATION 속성, $FILE_NAME 속성, $DATA 속성으로 나뉜다. 이 파일은 두 개의 $DATA 속성을 가지는데 실제 변경 로그 레코드를 저장하는 $J와 변경 로그의 기본 메타데이터를 저장하는 $Max가 그것이다.

9) 김진국, 'NTFS-소개' http://forensic-proof.com/archives/427, 2017.12.2.
10) Microsoft, 'Fsutil usn' http://technet.microsoft.com/en-us/library/cc788042.aspx, 2017.12.2.

	Offset	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Header	42457088	46	49	4C	45	30	00	03	00	A2	25	27	1A	00	00	00	00
	42457104	02	00	01	00	38	00	05	00	B0	01	00	00	00	04	00	00
	42457120	00	00	00	00	00	00	00	00	06	00	00	00	F6	A1	00	00
$STD_INFO	42457136	06	0F	47	11	00	00	00	00	10	00	00	00	60	00	00	00
	42457152	00	00	00	00	00	00	00	00	48	00	00	00	18	00	00	00
	42457168	5A	1B	46	FF	E1	E1	CA	01	5A	1B	46	FF	E1	E1	CA	01
	42457184	5A	1B	46	FF	E1	E1	CA	01	5A	1B	46	FF	E1	E1	CA	01
	42457200	26	02	00	00	00	00	00	00	00	00	00	00	00	00	00	00
	42457216	00	00	00	00	01	01	00	00	00	00	00	00	00	00	00	00
$FILE_NAME	42457232	00	00	00	00	00	00	00	00	30	00	00	00	70	00	00	00
	42457248	00	00	00	00	00	00	01	00	52	00	00	00	18	00	01	00
	42457264	0B	00	00	00	00	00	0B	00	5A	1B	46	FF	E1	E1	CA	01
	42457280	5A	1B	46	FF	E1	E1	CA	01	5A	1B	46	FF	E1	E1	CA	01
	42457296	5A	1B	46	FF	E1	E1	CA	01	00	00	00	00	00	00	00	00
$DATA/$J	42457312	00	00	00	00	00	00	00	00	26	00	00	00	00	00	00	00
	42457328	08	00	24	00	55	00	73	00	6E	00	4A	00	72	00	6E	00
	42457344	6C	00	00	00	00	00	00	00	80	00	00	00	60	00	00	00
	42457360	01	02	48	00	00	80	03	00	00	00	00	00	00	00	00	00
	42457376	6F	44	00	00	00	00	00	00	50	00	04	00	00	00	00	00
$DATA/$Max	42457392	00	00	47	04	00	00	00	00	18	6D	43	04	00	00	00	00
	42457408	18	6D	43	04	00	00	00	00	00	00	07	02	00	00	00	00
	42457424	24	00	4A	00	83	37	68	82	02	00	24	32	70	20	58	FE
	42457440	25	00	8E	01	A0	F8	FF	FF	80	00	00	00	40	00	00	00
	42457456	00	04	18	00	00	00	05	00	20	00	00	00	20	00	00	00
	42457472	24	00	4D	00	61	00	78	00	00	00	00	02	00	00	00	00
	42457488	00	00	40	00	00	00	00	00	5A	1B	46	FF	E1	E1	CA	01
	42457504	00	00	00	00	00	00	00	00	FF	FF	FF	FF	82	79	47	11

[그림3-2] $UsnJrnl의 MFT Entry (도구 : X-Ways Forensics v.19[11])

구분	헥스 값	변환 값
속성 타입 ID	0×00000080	128 ($DATA)
속성 길이	0×00000060	96
거주/비거주 속성	0×01	비거주
속성 이름 길이	0×02	2
속성 이름 오프셋	0×0048	72
플래그	0×8000	Sparse 속성
속성 ID(각 속성의 고유 값)	0×0003	3
런리스트 시작 VCN	0×000000000000	0
런리스트 마지막 VCN	0×00000000446F	17,519
런리스트 오프셋	0×0050	80
압축 유닛 크기	0×0004	16 클러스터
미사용	00 00 00 00	
속성 내용에 할당된 크기	0×0000000004470000	71,761,920
속성 내용의 실제 크기	0×0000000004436D18	71,527,704
속성 내용의 초기화된 크기	0×0000000004436D18	71,527,704

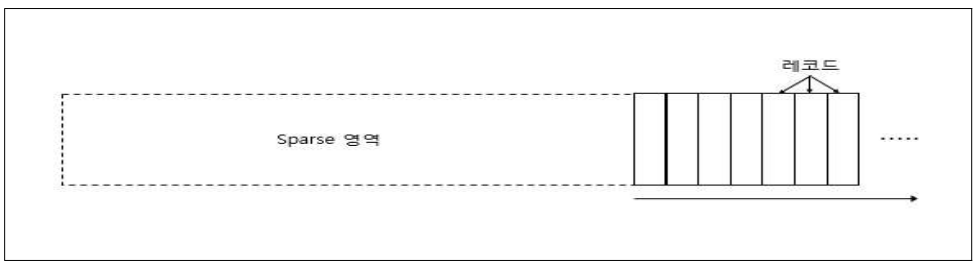
11) X-Ways Software Technology AG에서 개발한 디지털포렌식 분석도구이다.

미사용	00 00 07 02 00 00 00 00	
속성 이름	24 00 4A 00	$J
미사용	83 37 68 82	
클러스터 런리스트(Sparse)	02 00 24	시작 오프셋 : 0 길이 : 9,216 *9,216 × 4,096= 37,748,736
클러스터 런리스트(레코드)	32 70 20 58 FE 25 00 8E 01 A0 F8 FF FF	시작 오프셋 : 2,489,944 길이 : 8,304 *(9,216+8,304) × 4,096 = 71,761,920
미사용		

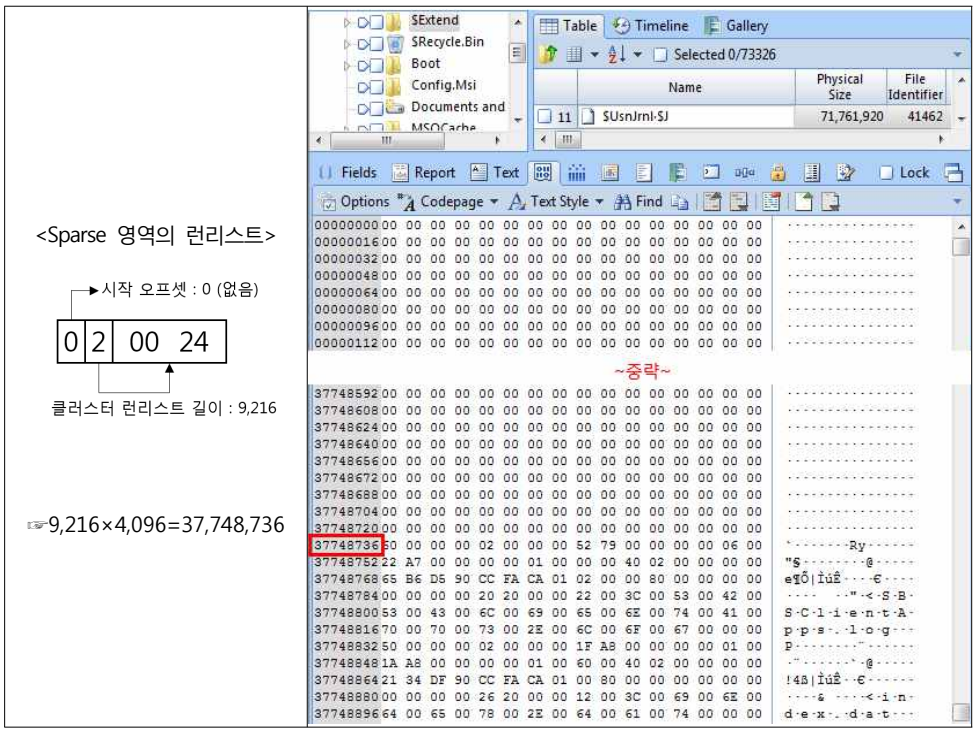
```
42457344                                80 00 00 00 60 00 00 00  l       €   `
42457360  01 02 48 00 00 80 03 00  00 00 00 00 00 00 00 00    H  €
42457376  6F 44 00 00 00 00 00 00  50 00 04 00 00 00 00 00  oD      P
42457392  00 00 47 04 00 00 00 00  18 6D 43 04 00 00 00 00    G      mC
42457408  18 6D 43 04 00 00 00 00  00 00 07 02 00 00 00 00  mC
42457424  24 00 4A 00 83 37 68 82  02 00 24 32 70 20 58 FE  $ J ƒ7h,  $2p Xþ
42457440  25 00 8E 01 A0 F8 FF FF                          % Ž  øÿÿ€   @
```

[표3-1] [그림3-2]의 $UsnJrnl·$J 데이터 속성 해석 (도구 : X-Ways Forensics v.19)

$UsnJrnl·$J의 속성 영역 중, [표3-1]의 클러스터 런리스트(Sparse)를 보면 시작 오프셋 값이 0인 것을 확인할 수 있다. 이 영역은 Sparse 영역을 뜻한다. Sparse란, 디스크 공간을 효율적으로 사용하기 위해 사용되지 않는 공간을 실제로 데이터를 덮어쓰지 않고 크기 정보만을 기록하는 것을 말한다. 따라서 레코드가 추가됨에 따라 파일의 크기는 계속 증가하지만 물리적인 크기는 일정하게 유지하게 된다. 이 Sparse 영역 뒤로 레코드들이 순차적으로 저장된다.



[그림3-3] $UsnJrnl 레코드 할당 방법



[그림3-4] $UsnJrnl:$J의 Sparse 영역 (도구 : EnCase v.7[12])

위의 [그림3-4]처럼 $UsnJrnl:$J의 Sparse 영역 뒤로 실제 변경 로그가 기록된 레코드가 이어진다. 이 레코드 안에는 변경 로그가 적용된 파일에 대한 정보가 기록되어 있는데, 객체 파일 참조 주소, 부모 파일 참조 주소, USN(Update Sequence Number), 변경 시간(UTC Time), 변경 원인, 파일 이름 등이 저장된다. 이 $UsnJrnl:$J에는 독립된 레코드 안에서 파일에 대한 시간 정보와 경로 정보를 얻을 수 있다는데 포렌식적 의미가 있다.

레코드의 파일 참조 주소는 MFT Entry 번호와 Sequence Value의 조합으로 이루어져 있다. 이 값은 MFT Entry의 변경 이력을 추적하는 의미있는 정보로 더 자세한 내용은 본 장 '3. 세 파일의 연결 원리'에서 다시 언급하도록 하겠다.

12) Guidance Software 사에서 개발한 디지털포렌식 분석 도구이다.

레코드 내 USN 값은 $UsnJrnl·$J의 오프셋 값과 일치하는데, 이는 이 레코드가 순차적으로 저장되는 것을 의미한다. 또한 이 USN 값은 변경 로그가 적용되는 파일의 MFT Entry 의 속성 정보 중$STANDARD_INFORMATION안에 기록된 USN과 일치한다. (단, MFT Entry에는 마지막 USN 번호만 기록한다.)

구분	헥스 값	변환 값
레코드 길이	0×00000060	96
Major 버전	0×0002	2
Miner 버전	0×0000	0
파일 참조 주소	0×000000007952 / 0×0006	31058, Sequence 번호 6
부모 파일 참조 주소	0×00000000A722 / 0×0001	42786, Sequence 번호 1
USN	0×0000000002400000	37,478,736
변경 시간	0×01CAFACC90D5B665	2010-5-24 08:06:24
변경 원인 플래그[13]	0×80000002	파일/디렉토리 닫힘, $DATA 속성에 데이터 추가
소스 정보[14]	00 00 00 00	0
보안 ID	00 00 00 00	0
파일 속성[15]	0×00002020	파일
파일 이름 길이	0×0022	34
파일 이름 오프셋	0×003C	60
파일 이름	53 00 43 00 6C 00 69 00 65 00 6E 00 74 00 41 00 70 00 70 00 73 00 2E 00 6C 00 6F 00 67	SBSClientApps.log

```
37748736 60 00 00 00 02 00 00 00 52 79 00 00 00 00 06 00 | `·····Ry·····
37748752 22 A7 00 00 00 00 01 00 00 00 40 02 00 00 00 00 | "§·····@·····
37748768 65 B6 D5 90 CC FA CA 01 02 00 00 80 00 00 00 00 | e¶Õ|ÌúÊ····€····
37748784 00 00 00 00 20 20 00 00 22 00 3C 00 53 00 42 00 | ····  ··"·<·S·B·
37748800 53 00 43 00 6C 00 69 00 65 00 6E 00 74 00 41 00 | S·C·l·i·e·n·t·A·
37748816 70 00 70 00 73 00 2E 00 6C 00 6F 00 67 00 00 00 | p·p·s·.·l·o·g···
```

[표3-2] [그림3-3]의 $UsnJrnl·$J의 첫 번째 레코드 정보

13) [별지1]의 변경 원인 플래그(Reason Flag) 참고
14) [별지2]의 소스 정보(Source Information) 참고
15) [별지3]의 파일 속성(File Attribute) 참고

$UsnJrnl·$Max에는 변경 로그의 기본 메타데이터를 저장한다. 이 파일은 거주 속성으로 속성 헤더 뒤에 데이터를 가지고 있다.

구분	헥스 값	변환 값
속성 타입 ID	0×00000080	128($DATA)
속성 길이	0×00000040	64
거주/비거주 속성	0×00	거주
속성 이름 길이	0×04	4
속성 이름 오프셋	0×0018	24
플래그	0×0000	0
속성ID(각 속성의 고유 값)	0×0005	5
속성 내용 크기	0×00000020	32
속성 내용 오프셋	0×0020	32
미사용	0000	
속성 이름	24 00 4D 00 61 00 78 00	$MAX
속성 내용	00 00 00 02 00 00 00 00 00 00 40 00 00 00 00 00 5A 1B 46 FF E1 E1 CA 01 00 00 00 00 00 00 00 00	$UsnJrnl 파일의 최대 사이즈 : 33,554,432 할당/할당해제에 사용된 바이트 수 : 4,194,304 USN ID 최저 유효 USN

```
42457440                                80 00 00 00 40 00 00 00  € Ž  øÿÿ€    @
42457456  00 04 18 00 00 00 05 00  20 00 00 00 20 00 00 00
42457472  24 00 4D 00 61 00 78 00  00 00 00 02 00 00 00 00  $ M a x
42457488  00 00 40 00 00 00 00 00  5A 1B 46 FF E1 E1 CA 01    @      Z FÿááÊ
42457504  00 00 00 00 00 00 00 00  FF FF FF FF 82 79 47 11          ÿÿÿÿ,yG
```

[표3-3] [그림3-2]의 $UsnJrnl·$Max 속성 헤더의 해석

② 변경 원인 플래그

$UsnJrnl은 변경 여부 및 변경 사유만을 기록하는데 이러한 변경 사유를 식별하기 위해 변경 원인 플래그를 사용한다. 예를 들어 파일을 생성하는 경우 NTFS 파일 시스템은 'FILE_CREATE(0x00000100)' 라는 변경 원인 플래그를 사용하여 레코드를 기록하고, 파일이 삭제되는 경우 'FILE_DELETE(0x00000200)' 라는 변경 원인 플래그를 레코드에 기록하는 형식이다. 사용자가 인식하는 하나의 이벤트는 실제 시스템 안에서는

여러 개의 작업으로 이루어진다. 예를 들면, Delete 키를 이용하여 파일을 삭제하는 경우 세 개의 레코드 각각에 아래와 같은 변경 원인 플래그를 기록한다.

RENAME_OLD_NAME /0x00001000 RENAME_NEW_NAME /0x00002000 CLOSE /0x80000000

일반적으로 파일을 삭제하는 경우 휴지통으로 이동하면서 파일명이 변경된다. 따라서 변경 원인 플래그에서는 객체명을 변경하는 레코드를 기록하게 되는 것이다. 마지막 레코드의 CLOSE는 파일이 닫히는 경우 생성되는 최종 레코드로 해당 작업이 끝났음을 의미한다.

만약 동일한 파일을 여러번 변경하면 현재 레코드에 하나의 변경 원인 플래그만 추가될 수 있는데, 이는 같은 종류의 변경이 두 번 이상 발생하면 NTFS 파일 시스템은 첫 번째 이후에 변경 내용에 대한 새 레코드를 작성하지는 않기 때문이다. 즉 파일을 작업하는 과정에서 파일을 닫지 않는 한 여러 번의 파일 내용 쓰기 작업은 'DATA_OVERWRITE /0x00000001'가 설정된 변경 레코드 하나만 가져오게 된다.[16]

③ $UsnJrnl의 할당 방법

$UsnJrnl 파일의 레코드는 순차적으로 저장된다. 만약 고정된 속성 사이즈가 모두 차면 오래된 레코드를 덮어쓰지 않고 새로운 영역을 할당받아 레코드를 생성한다. 단, $UsnJrnl·$Max에서 정해진 할당 사이즈를 유지하기 위해 가장 오래된 레코드 영역에 대해서는 할당을 해제하고 Sparse 영역으로 관리하게 된다. 비할당 영역으로 바뀐 Sparse 영역은 실제 0으로 채워지는 것은 아니고 크기 정보만 기록한다. 따라서 파일의

16) Microsoft, 'Change Journal', https://msdn.microsoft.com/en-us/library/windows/desktop/aa363798(v=vs.85).aspx, 2017.12.2.

논리적인 크기는 실제 물리적으로 할당된 양보다 크다. 이렇게 Sparse 영역으로 채워지면 기존의 레코드들은 비할당 영역에 존재하게 된다. 따라서 비할당 영역을 대상으로 카빙을 시도하면 상당한 양의 레코드들이 복구되는 것을 알 수 있다.

아래의 [그림3-5]는 실제 HDD 분석 사례에서 비할당 영역을 대상으로 $UsnJrnl 레코드를 카빙한 결과이다. 카빙 결과 약 250,751KB의 파일(1,295,493개의 레코드)이 복구 되었고, 해당 레코드 내 정보를 확인해 보면 2013년 11월 19일 경부터의 레코드들이 복구된 것을 확인할 수 있다. 이는 실제 할당 영역의 가장 오래된 레코드가 2017년 1월 23일인 것과 비교했을 때 약 4년 전의 기록이다.

Offset	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	Time Stamp
000000000	50	00	00	00	02	00	00	00	5D	00	00	00	00	00	01	00	2013-11-19 10:56:11(+9)
000000016	23	00	00	00	00	00	01	00	00	00	04	00	00	00	00	00	
000000032	B8	2F	B5	F5	15	E5	CE	01	01	00	00	80	00	00	00	00	
000000048	00	00	00	00	26	00	00	00	0E	00	3C	00	42	00	43	00	
000000064	44	00	2E	00	4C	00	4F	00	47	00	00	00	00	00	8B	C7	

비할당 영역의 $UsnJrnl 레코드 일부 (가장 오래된 레코드)

Offset	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	Time Stamp
14680064	50	00	00	00	02	00	00	00	FA	03	00	00	00	00	0E	00	2017-01-23 00:35:37(+9)
14680080	F3	03	00	00	00	00	03	00	00	00	E0	00	00	00	00	00	
14680096	3C	A6	DF	9D	10	75	D2	01	00	20	00	80	00	00	00	00	
14680112	00	00	00	00	20	00	00	00	12	00	3C	00	31	00	6E	00	
14680128	70	00	67	00	67	00	2E	00	65	00	72	00	6C	00	00	00	

할당 영역의 $UsnJrnl 레코드 일부 (가장 오래된 레코드)

<하드디스크 정보>
Label : Samsung
Model : HD502HJ
Total Size : 465.8GB
Partition Part1 : NTFS, 341.8GB / Part2 : NTFS, 124GB
포맷일시 : 2012-11-29 17:14:21

[그림3-5] 비할당 영역에서 카빙한 $UsnJrnl 레코드

비할당 영역에서 카빙한 파일들의 경우 MFT 영역에서 복구된 데이터가 아니므로 실제 해당 복구 파일들은 신뢰성을 확보하기 위해 몇 가지 확인해야 하는 정보가 있다.

만일 사용자가 해당 디스크의 최초 사용자였다는 사실이 전제가 된다

면 복구되는 레코드는 사용자의 흔적일 가능성이 크고, 포맷하는 경우 레코드가 다시 시작 되므로 하드 드라이브의 포맷 일시도 함께 확인해 볼 필요가 있다. 또한 볼륨 마다 $UsnJrnl 파일이 존재하는 경우라면 카빙 작업 시 볼륨의 구분도 필요하다. 비할당 영역의 레코드들은 순차적으로 레코드가 생성됨을 감안할 때 각 레코드의 USN 번호를 통해 행위의 순서를 확인해 볼 수 있다.

위의 비할당 영역에서의 복구된 레코드의 양은 사용자에 따라, 하드디스크의 종류에 따라 차이가 있다. SSD의 경우 TRIM 작업에 의해 비할당 영역이 시스템에 의해 정리되기 때문에 비할당 영역에 남아있는 데이터가 적다. 실제 위의 HDD를 대상으로 복구한 결과 데이터가 약 4년 전의 기록까지 복구된 것과 비교했을 때 상대적으로 SSD를 대상으로 카빙을 시도했을 때에는 비할당 영역에서 카빙된 데이터는 288,891KB로 약 12분 동안의 레코드만이 복구되었다.

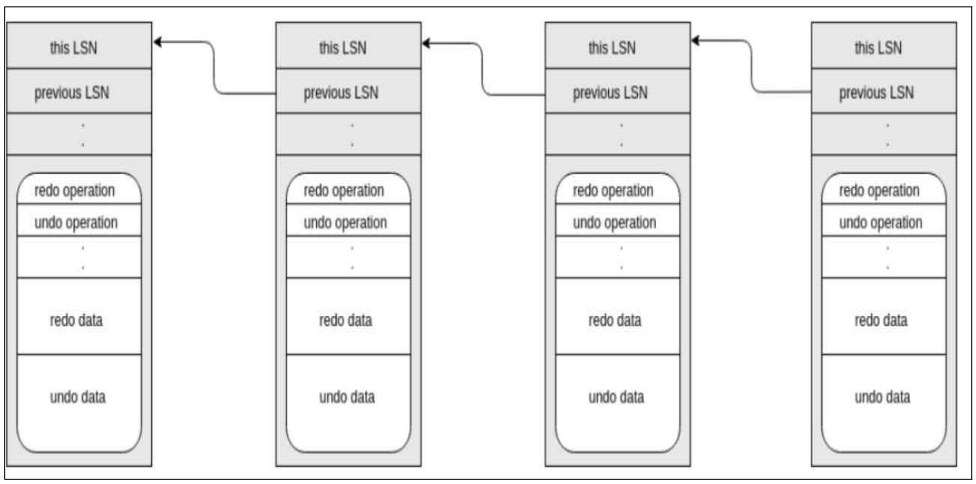
(3) $LogFile

NTFS 파일 시스템의 저널 기능과 관련된 정보를 저장하는 또 다른 파일이 $LogFile이다.[17] 이 파일은 데이터의 신뢰성을 확보하기 위해 완성되지 않은 작업은 업데이트 전의 상태로 되돌린다. 이전 상태로 되돌리기 위해 이 $LogFile은 변경된 정보(Redo)와 변경 직전(Undo)의 정보를 모두 가지고 있게 된다.

이러한 변경 전후의 정보는 $LogFile을 구성하는 레코드 안에 존재하는데, 이 레코드들이 여러 개 모여 하나의 트랜잭션 단위를 이룬다. 시스템은 모든 작업을 트랜잭션 단위로 관리하는데, 만약 이 트랜잭션이 완성되지 못하면 레코드 Undo 정보를 통해 완성 전으로 되돌리는 원리이다.

---

17) Brian Carrier, 앞의 책, 340-341



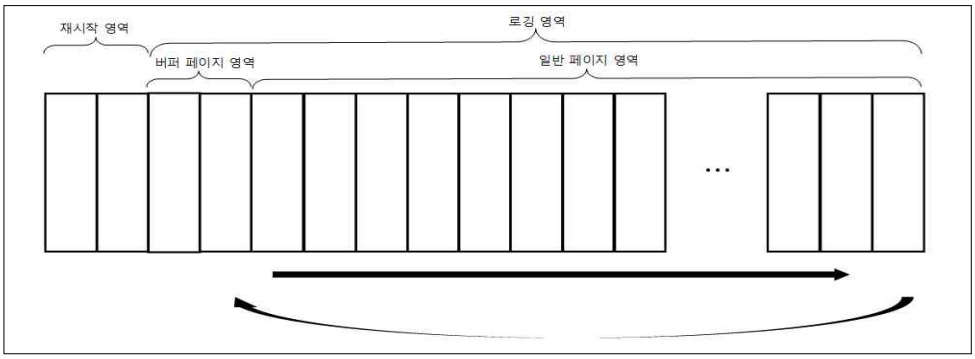
[그림3-6] $LogFile 레코드 연결[18)]

$LogFile은 크게 두 개의 영역으로 나뉜다. 재시작 영역과 로깅 영역이다. 각 영역은 페이지 단위로 구성되어 있고, 각 페이지는 레코드들로 구성된다. 재시작 영역은 가장 마지막 작업 레코드를 가지고 있으면서 앞으로의 작업 레코드의 시작 위치를 알려주는 역할을 하고, 로깅 영역은 실제 작업 레코드들이 기록되는데, 이 로깅 영역은 다시 버퍼 페이지 영역과 일반 페이지 영역으로 나누어진다. 버퍼 페이지인 첫 두 페이지 외의 일반 페이지 영역에는 작업 레코드들이 순차적으로 저장된다. 만약 로깅 영역이 모두 차면 가장 오래된 레코드들부터 차례로 덮어쓰게 된다.[19)]

각각의 레코드들은 64비트 값의 LSN($LogFile Sequence Number)을 가지고 있고, 만약 할당된 크기가 가득 차면 가장 오래된 레코드 위에 새로운 레코드를 기록하게 된다. 따라서 로그 파일의 시작 레코드가 가장 마지막 레코드보다 크게 되는 경우가 있는데 이는 레코드가 위치에 상관없이 순차적으로 생성됨을 의미하는 것이다.

---

18) Frank Uijtewaal 외 1, 앞의 논문, 3, Figure 1.
19) 오정훈, 'F-INSIGHT-NTFS Log Tracker' FORENSIC INSIGHT ; DIGITAL FORENSICS COMMUNITY IN KOREA(2013), 3



[그림3-7] $LogFile 레코드 할당 방법

위 파일에는 변경되는 파일의 정보를 가지고 있는데, MFT Entry 속성 중 $STANDARD_INFORMATION로부터 시간 정보를, $FILE_NAME으로 부터 파일 이름과 부모 파일 참조 주소를 가져온다.

$LogFile은 MFT Entry 2번에 고정적으로 할당되어 있고, 이 MFT Entry는 $DATA 속성 안에 로그 데이터를 저장한다. 이 파일은 각 볼륨마다 하나씩 존재하고 일반적인 하드디스크 볼륨에서는 64MB의 크기를 가지고 있으며 아래의 명령어를 통해 사용자가 그 크기를 조절할 수 있다.

chkdsk /l:<파일크기(KB)>

모든 MFT Entry에는 특정 이벤트 트랜잭션의 마지막 LSN이 기록된다. $LogFile의 Target VCN(Virtual Cluster Number, 대상 파일의 논리적 클러스터 번호)과 MFT Cluster Index(MFT Entry가 있는 하나의 클러스터 내에서의 Entry 위치[20])를 아래의 계산식으로 계산하면 MFT Entry 번호를 확인할 수 있다.

20) 오정훈, 앞의 자료, 18

$$MFT\ entry\ number = VCN \times 4 + \frac{mft\ cluster\ index}{2}$$

[그림3-8] MFT Entry 번호 계산식[21)]

0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
This LSN								Previous LSN							
Client Undo LSN								Client Data Length				Client Id			
Record Type				Transaction ID				Flags		Alignment or Reserved					
Redo OP		Undo OP		Redo Offset		Redo Length		Undo Offset		Undo Length		Target Attr		LCNs to follows	
Record Offset		Attr Offset		MFT Cluster INDX		Alignment or Reserved		Target VCN				Alignment or Reserved			
Target LCN				Alignment or Reserved											

[그림3-9] $LogFile의 레코드 구조

$LogFile은 $UsnJrnl 파일과 달리 최대 용량을 넘어서는 경우, 가장 오래된 LSN 위치로 돌아와 덮어쓰여진다. 따라서 과거의 레코드가 삭제되면 비할당 영역에 존재하지 않는다.

2. 포렌식 관점에서의 두 저널파일 비교

$LogFile의 경우 업데이트가 되기 전(변경 전)의 메타데이터 정보와 업데이트 된(변경 후) 메타데이터 정보를 모두 기록하기 때문에 데이터 확보에 유리하다. 반면 $UsnJrnl 파일은 변경 여부와 변경 정보에 대한 기록만을 가지기 때문에 데이터 복구는 불가능하다.

21) Frank Uijtewaal 외 1, 앞의 논문 11, Appendix

기본적으로 $LogFile은 약 65MB의 사이즈를 가지는데 이는 하루에 8시간 정도 사용할 경우 약 2~3 시간 가량의 로그가 기록[22]될 수 있는 양이다. 반면 $UsnJrnl 파일은 보통 34MB를 최대 로그 사이즈를 가지며 하루 8시간 사용하는 경우 4~5일 정도의 로그가 남는다. 사이즈로는 $LogFile이 크지만 $LogFile이 트랜잭션 단위로 이벤트를 기록하기 때문에 약 1~2개의 레코드 만을 사용하는 $UsnJrnl 파일에 비해 기록되는 이벤트 수는 적다.

디지털포렌식 분석의 활용도 측면에서 볼 때 최근의 이벤트로서 자세한 정보나 데이터 복구가 필요한 사안이라면 $LogFile을 활용해 볼 수 있겠고, 전반적인 이벤트 변화 정보나 단편적인 변경 정보를 얻어야 하는 경우라면 $UsnJrnl 파일을 활용해 볼 수 있겠다.

구분	$UsnJrnl	$LogFile
기록 정보	파일/ 디렉토리 변경 원인 정보 기록	파일/디렉토리 변경 이전/이후 상태 정보의 기록
데이터 복구 가능 여부	불가능	가능
위치(Default)	운영체제 볼륨에만 존재	모든 볼륨에 존재
논리적 파일 크기	계속 증가	고정
비할당 영역	이전 레코드 기록 존재	이전 레코드 기록 없음
파일에 할당된 클러스터 수	(Maximum size+allocation delta)/Cluster 크기 내에서 가변적	불변

[표3-4] $UsnJrnl파일과 $LogFile 비교

22) 오정훈, 앞의 자료, 45

3. 세 파일의 연결 원리

시스템 내에서 사용자나 시스템이 하나의 이벤트를 발생시키면 시스템은 그 파일을 관리하는 MFT Entry를 새로 할당하고 $UsnJrnl 파일과 $LogFile에 발생한 이벤트를 로그로 기록한다. 이때 $UsnJrnl 파일과 $LogFile은 대상 파일에 대한 정보를 MFT Entry에서 가져오기 때문에 두 파일은 MFT Entry 정보를 포함하고 있다. 이때 사용되는 정보가 MFT Entry 번호이다.

MFT Entry 번호는 해당 Entry의 시작 오프셋의 45번째 바이트부터 4바이트가 MFT Entry 번호를 나타내므로 이를 통해 번호를 알아내거나, 해당 번호는 오프셋 위치에서부터 1,024 바이트 단위로 0번에서부터 순차적으로 정의되므로 할당 영역의 $MFT 파일인 경우 파일 오프셋 값을 1,024 바이트로 나누어도 해당 MFT Entry의 번호를 알 수 있다. $LogFile은 Target VCN과 MFT Cluster Index 정보를 통해 그 번호를 알 수 있는데 $LogFile의 [그림3-7]의 공식을 이용하면 MFT Entry 번호를 산출해 낼 수 있다. 마지막으로 $UsnJrnl파일은 파일 참조 주소의 앞 6바이트가 MFT Entry 번호를 나타내므로 이를 통해 그 값을 알 수 있다.

아래의 그림은 MFT Entry 번호로 연결되는 세 파일의 관계를 확인하기 위해 별도의 실험을 진행한 결과이다. 임의로 생성한 파일인 '1sttest.txt'은 5292번의 MFT Entry 번호를 할당 받아 사용하였고 이 파일의 생성이벤트를 기록한 $UsnJrnl 파일과 $LogFile은 MFT Entry 번호 '5292'번으로 연결된다.

파일명 : 1sttest.txt

이벤트 : 파일 생성

	Name	File Ext	File Identifier	Entry Modified	File Created	Last Written	Last Accessed
32	1sttest.txt	txt	5292	2017-11-26 17:40:37	2017-11-26 17:40:36	2017-11-26 17:40:37	2017-11-26 17:40:36

구분	헥스 값	MFT Entry 번호
1sttest.txt의 MFT Entry	Offset 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 005419008 46 49 4C 45 30 00 03 00 97 25 08 23 0F 00 00 00 005419024 1E 00 01 00 38 00 01 00 30 01 00 00 00 04 00 00 005419040 00 00 00 00 00 00 00 00 03 00 00 00 AC 14 00 00 005419056 02 00 00 00 00 00 00 00 10 00 00 00 60 00 00 00 005419072 00 00 00 00 00 00 00 00 48 00 00 00 18 00 00 00 005419088 BD FF FC 3A 92 66 D3 01 96 49 5D 3B 92 66 D3 01 005419104 96 49 5D 3B 92 66 D3 01 BD FF FC 3A 92 66 D3 01 005419120 20 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 005419136 00 00 00 00 15 04 00 00 00 00 00 00 00 00 00 00 005419152 68 B7 0B 97 04 00 00 00 30 00 00 00 70 00 00 00 005419168 00 00 00 00 00 00 02 00 58 00 00 00 18 00 01 00 005419184 B1 08 00 00 00 00 04 00 BD FF FC 3A 92 66 D3 01 005419200 BD FF FC 3A 92 66 D3 01 BD FF FC 3A 92 66 D3 01 005419216 BD FF FC 3A 92 66 D3 01 00 00 00 00 00 00 00 00 005419232 00 00 00 00 00 00 00 00 20 00 00 00 00 00 00 00 005419248 0B 03 31 00 73 00 74 00 74 00 65 00 73 00 74 00 005419264 2E 00 74 00 78 00 74 00 80 00 00 00 20 00 00 00 005419280 00 00 18 00 00 00 01 00 02 00 00 00 18 00 00 00 005419296 0D 0A FF FF 82 79 47 11 FF FF FF FF 82 79 47 11	· MFT Entry 번호 : **5292** (또는 file offset / 1024 = 5292) · LSN : 0×0000000F23082597 · USN : 0×00000004970BB768
$LogFile 레코드 (이벤트의 마지막 레코드)	Offset 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 04271280 00 00 1B 02 00 00 00 00 97 25 08 23 0F 00 00 00 04271296 84 25 08 23 0F 00 00 00 84 25 08 23 0F 00 00 00 04271312 38 00 00 00 00 00 00 00 01 00 00 00 40 00 00 00 04271328 00 00 00 00 00 00 00 00 07 00 07 00 28 00 08 00 04271344 30 00 08 00 18 00 01 00 38 00 58 00 00 00 02 00 04271360 2B 05 00 00 00 00 00 00 2B 05 0C 00 00 00 00 00	· MFT Entry 번호 : VCN×4+MFT cluster index/2 = **5292** · LSN : 0×0000000F23082597
$UsnJrnl 레코드 (이벤트의 마지막 레코드)	Offset 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 19713996640 74 00 00 00 00 00 00 00 58 00 00 00 02 00 00 00 19713996656 AC 14 00 00 00 00 1E 00 B1 08 00 00 00 00 04 00 19713996672 68 B7 0B 97 04 00 00 00 AF 8A 0A 3D 92 66 D3 01 19713996688 02 01 00 80 00 00 00 00 00 00 00 00 20 00 00 00 19713996704 16 00 3C 00 31 00 73 00 74 00 74 00 65 00 73 00 19713996720 74 00 2E 00 74 00 78 00 74 00 00 00 00 00 00 00	· MFT Entry 번호 : **5292** · USN : 0×00000004970BB768

[그림3-10] MFT Entry 5292번으로 연결되는 세 파일(레코드) (도구 : EnCase v.7)

위의 그림에서처럼 MFT Entry 번호로 연결해 보면 MFT Entry에는 관련된 $LogFile과 $UsnJrnl 파일의 LSN 정보와 USN 정보를 모두 가지고 있는 것을 확인할 수 있다. 하나의 이벤트는 보통 하나 이상의 레코드로 이루어지는데 이 중 MFT Entry에는 가장 마지막 LSN과 USN만을 기록한다.

4. $UsnJrnl 파일을 통한 MFT Entry 정보 분석

$UsnJrnl 파일의 각 레코드는 이벤트가 발생하는 파일의 이름, 시간 정보, 전체 경로 등의 정보를 포함한다. 아래의 그림은 '1sttest.txt' 파일의 MFT Entry 정보와 이 파일을 생성한 $UsnJrnl 레코드를 비교한 그림이다. 그림을 통해 각 레코드에 포함된 객체 파일의 정보가 MFT Entry에 기록된 객체 파일의 정보가 일치함을 알 수 있다. 즉 MFT Entry 정보 없이도 $UsnJrnl 레코드를 통해서 객체 파일의 일부 정보를 알 수 있음을 의미한다.

<table>
<tr><th>구분</th><th>헥스 값</th><th>오프셋 정보</th></tr>
<tr><td>1sttest.txt의
MFT Entry</td><td>
Offset 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15
005419008 46 49 4C 45 30 00 03 00 97 25 08 23 0F 00 00 00
005419024 1E 00 01 00 38 00 01 00 30 01 00 00 00 04 00 00
005419040 00 00 00 00 00 00 00 00 03 00 00 00 AC 14 00 00
005419056 02 00 00 00 00 00 00 00 10 00 00 00 60 00 00 00
005419072 00 00 00 00 00 00 00 00 48 00 00 00 18 00 00 00
005419088 BD FF FC 3A 92 66 D3 01 96 49 5D 3B 92 66 D3 01
005419104 96 49 5D 3B 92 66 D3 01 BD FF FC 3A 92 66 D3 01
005419120 20 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
005419136 00 00 00 00 15 04 00 00 00 00 00 00 00 00 00 00
005419152 68 B7 0B 97 04 00 00 00 30 00 00 00 70 00 00 00
005419168 00 00 00 00 00 00 02 00 58 00 00 00 18 00 01 00
005419184 B1 08 00 00 00 00 04 00 BD FF FC 3A 92 66 D3 01
005419200 BD FF FC 3A 92 66 D3 01 BD FF FC 3A 92 66 D3 01
005419216 BD FF FC 3A 92 66 D3 01 00 00 00 00 00 00 00 00
005419232 00 00 00 00 00 00 00 00 20 00 00 00 00 00 00 00
005419248 0B 03 31 00 73 00 74 00 74 00 65 00 73 00 74 00
005419264 2E 00 74 00 78 00 74 00 80 00 00 00 20 00 00 00
005419280 00 00 18 00 00 00 01 00 02 00 00 00 18 00 00 00
005419296 0D 0A FF FF 82 79 47 11 FF FF FF FF 82 79 47 11
</td><td>
· USN : 0×00000004970BB768
· 파일생성시간 : 17-11-26 17:40:36 (0×013D66923AFCFFBD)
· 부모 MFT 파일 참조 주소 : 0×0004000000008B1
· 파일명 : 1sttest.txt
</td></tr>
<tr><td>$UsnJrnl
레코드</td><td>
Offset 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15
19713996464 6C 00 00 00 00 00 00 05 58 00 00 00 02 00 00 00
19713996480 AC 14 00 00 00 00 1E 00 B1 08 00 00 00 00 04 00
19713996496 B8 B6 0B 97 04 00 00 00 BD FF FC 3A 92 66 D3 01
19713996512 00 01 00 00 00 00 00 00 00 00 00 00 20 00 00 00
19713996528 16 00 3C 00 31 00 73 00 74 00 74 00 65 00 73 00
19713996544 74 00 2E 00 74 00 78 00 74 00 00 00 00 00 00 00
19713996560 58 00 00 00 02 00 00 00 AC 14 00 00 00 00 1E 00
19713996576 B1 08 00 00 00 00 04 00 10 B7 0B 97 04 00 00 00
19713996592 BD FF FC 3A 92 66 D3 01 02 01 00 00 00 00 00 00
19713996608 00 00 00 00 20 00 00 00 16 00 3C 00 31 00 73 00
19713996624 74 00 74 00 65 00 73 00 74 00 2E 00 74 00 78 00
19713996640 74 00 00 00 00 00 00 00 58 00 00 00 02 00 00 00
19713996656 AC 14 00 00 00 00 1E 00 B1 08 00 00 00 00 04 00
19713996672 68 B7 0B 97 04 00 00 00 AF 8A 0A 3D 92 66 D3 01
19713996688 02 01 00 80 00 00 00 00 00 00 00 00 20 00 00 00
19713996704 16 00 3C 00 31 00 73 00 74 00 74 00 65 00 73 00
19713996720 74 00 2E 00 74 00 78 00 74 00 00 00 00 00 00 00
</td><td>
· USN : 0×00000004970BB6B8
· 기록 시간 : 17-11-26 17:40:36 (0×01D366923AFCFFBD)
· 파일명 : 1sttest.txt

· USN : 0×00000004970BB710
· 기록 시간 : 17-11-26 17:40:36 (0×01D366923AFCFFBD)
· 파일명 : 1sttest.txt

· USN : 0×00000004970BB768
· 기록 시간 : 17-11-26 17:40:39 (0×01D366923D0A8AAF)
· 파일명 : 1sttest.txt
</td></tr>
</table>

[그림3-11] MFT Entry와 $UsnJrnl 레코드 비교

MFT Entry는 해당 Entry를 재할당해 다시 사용하는 반면 $UsnJrnl 파일은 해당 레코드를 재사용하는 것이 아니라 새로운 레코드를 할당받아 기록한다. 따라서 MFT Entry의 경우 재할당되면 덮어쓰이기 전의 정보를 알 수 없지만 $UsnJrnl 파일 레코드는 비할당영역으로 바뀌더라도 덮어쓰이지 않는 한 파일에 대한 정보를 그대로 가지고 있게 된다. 따라서 삭제되어 덮어쓰여진 MFT Entry 정보를 이 $UsnJrnl 파일 레코드로부터 확인이 가능하다.

아래의 [그림3-12]은 MFT Entry가 덮어쓰이기 전과 덮어쓰인 이후의 $UsnJrnl파일 레코드들을 비교한 것이다.

'1sttest.txt' 파일은 5292번의 MFT Entry 번호를 사용하다가 이후 삭제되었고 해당 5292번의 MFT Entry는 '2ndtest.txt' 파일의 MFT Entry로 재할당되어 사용하게 된다. 이때 $UsnJrnl 파일을 살펴 보면 '1sttest.txt' 와 관련된 $UsnJrnl 레코드의 USN 번호는 '19713996472', '19713996560', '19713996648' 이고, '2ndtest.txt' 파일과 관련된 $UsnJrnl 레코드의 USN 번호는 '19734692320', '19734692408', '19734694200'으로 그 값이 증가하는 것을 알 수 있다.

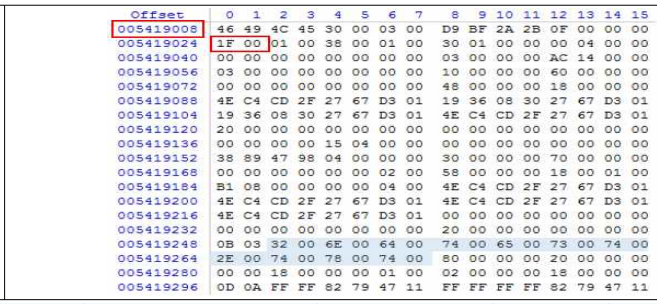
1sttest.txt MFT Entry

Offset	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
005419008	46	49	4C	45	30	00	03	00	97	25	08	23	0F	00	00	00
005419024	1E	00	01	00	38	00	01	00	30	01	00	00	00	04	00	00
005419040	00	00	00	00	00	00	00	00	03	00	00	00	AC	14	00	00
005419056	02	00	00	00	00	00	00	00	10	00	00	00	60	00	00	00
005419072	00	00	00	00	00	00	00	00	48	00	00	00	18	00	00	00
005419088	BD	FF	FC	3A	92	66	D3	01	96	49	5D	3B	92	66	D3	01
005419104	96	49	5D	3B	92	66	D3	01	BD	FF	FC	3A	92	66	D3	01
005419120	20	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
005419136	00	00	00	00	15	04	00	00	00	00	00	00	00	00	00	00
005419152	68	B7	0B	97	04	00	00	00	30	00	00	00	70	00	00	00
005419168	00	00	00	00	00	00	02	00	58	00	00	00	18	00	01	00
005419184	B1	08	00	00	00	00	04	00	BD	FF	FC	3A	92	66	D3	01
005419200	BD	FF	FC	3A	92	66	D3	01	BD	FF	FC	3A	92	66	D3	01
005419216	BD	FF	FC	3A	92	66	D3	01	00	00	00	00	00	00	00	00
005419232	00	00	00	00	00	00	00	00	20	00	00	00	00	00	00	00
005419248	0B	03	31	00	73	00	74	00	74	00	65	00	73	00	74	00
005419264	2E	00	74	00	78	00	74	00	80	00	00	00	20	00	00	00
005419280	00	00	18	00	00	00	01	00	02	00	00	00	18	00	00	00
005419296	0D	0A	FF	FF	82	79	47	11	FF	FF	FF	FF	82	79	47	11

USN	레코드 시간 정보	변경 원인 플래그	대상 파일 이름	대상 파일의 MFT Entry 번호	대상 파일의 MFT Entry Sequence 번호
19713996472	2017-11-26 17:40:36	FILE_CREATE(0x00000100)	1sttest.txt	5292	30
19713996560	2017-11-26 17:40:36	DATA_EXTEND(0x00000002) FILE_CREATE(0x00000100)	1sttest.txt	5292	30
19713996648	2017-11-26 17:40:39	DATA_EXTEND(0x00000002) FILE_CREATE(0x00000100) CLOSE(0x80000000)	1sttest.txt	5292	30

<덮어쓰이기 전>

2ndtest.txt MFT Entry



Offset	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
005419008	46	49	4C	45	30	00	03	00	D9	BF	2A	2B	0F	00	00	00
005419024	1F	00	01	00	38	00	01	00	30	01	00	00	00	04	00	00
005419040	00	00	00	00	00	00	00	00	03	00	00	00	AC	14	00	00
005419056	03	00	00	00	00	00	00	00	10	00	00	00	60	00	00	00
005419072	00	00	00	00	00	00	00	00	48	00	00	00	18	00	00	00
005419088	4E	C4	CD	2F	27	67	D3	01	19	36	08	30	27	67	D3	01
005419104	19	36	08	30	27	67	D3	01	4E	C4	CD	2F	27	67	D3	01
005419120	20	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
005419136	00	00	00	00	15	04	00	00	00	00	00	00	00	00	00	00
005419152	38	89	47	98	04	00	00	00	30	00	00	00	70	00	00	00
005419168	00	00	00	00	00	00	02	00	58	00	00	00	18	00	01	00
005419184	B1	08	00	00	00	00	04	00	4E	C4	CD	2F	27	67	D3	01
005419200	4E	C4	CD	2F	27	67	D3	01	4E	C4	CD	2F	27	67	D3	01
005419216	4E	C4	CD	2F	27	67	D3	01	00	00	00	00	00	00	00	00
005419232	00	00	00	00	00	00	00	00	20	00	00	00	00	00	00	00
005419248	0B	03	32	00	6E	00	64	00	74	00	65	00	73	00	74	00
005419264	2E	00	74	00	78	00	74	00	80	00	00	00	20	00	00	00
005419280	00	00	18	00	00	00	01	00	02	00	00	00	18	00	00	00
005419296	0D	0A	FF	FF	82	79	47	11	FF	FF	FF	FF	82	79	47	11

USN	레코드 시간 정보	변경 원인 플래그	대상 파일 이름	대상 파일의 MFT Entry 번호	대상 파일의 MFT Entry Sequence 번호
19734692320	2017-11-27 11:26:52	FILE_CREATE(0x00000100)	2ndtest.txt	5292	31
19734692408	2017-11-27 11:26:52	DATA_EXTEND(0x00000002) FILE_CREATE(0x00000100)	2ndtest.txt	5292	31
19734694200	2017-11-27 11:27:08	DATA_EXTEND(0x00000002) FILE_CREATE(0x00000100) CLOSE(0x80000000)	2ndtest.txt	5292	31

<덮어쓰인 후>

[그림3-12] 덮어쓰이기 전과 후의 MFT Entry와 $UsnJrnl 레코드 비교

아래의 [그림3-13]은 $UsnJrnl 파일 중 5292번의 MFT Entry 번호를 사용하는 파일들을 그 객체로 하는 레코드들 중 일부이다. 레코드①은 '1sttest.txt'가 생성된 레코드이고, 레코드②은 '1sttest.txt'가 삭제된 레코드이며, 레코드③은 '2ndtest.txt'가 생성된 레코드이다. 즉 '2ndtest.txt'파일의 MFT Entry는 '1sttest.txt' 가 사용하던 MFT Entry를 재할당해 사용하면서 '1sttest.txt' 가 사용하던 MFT Entry를 덮어쓴 경우다. MFT Entry 만으로는 '1sttest.txt'에 관한 정보를 알 수 없으나 이 $UsnJrnl의 레코드들로부터 덮어쓰기 전의 파일이 '1sttest.txt'라는 것을 포함해 해당 파일의 시간정보, 전체 경로 정보를 알 수 있게 된다. 또한 대상 파일의 MFT Entry Sequence 번호가 30에서 31로 증가한 것을 확인할 수 있는데, 이는 '1sttest.txt' 파일에 할당된 후 다른 파일에 재할당되지 않고 '2ndtest.txt'파일에 할당되었음을 의미한다.

	USN	레코드 시간 정보	변경 원인 플래그	대상 파일 이름	대상 파일의 MFT Entry 번호	대상 파일의 MFT Entry Sequence 번호
①	19713996472	2017-11-26 17:40:36	FILE_CREATE(0x00000100)	1sttest.txt	5292	30
	19713996560	2017-11-26 17:40:36	DATA_EXTEND(0x00000002) FILE_CREATE(0x00000100)	1sttest.txt	5292	30
	19713996648	2017-11-26 17:40:39	DATA_EXTEND(0x00000002) FILE_CREATE(0x00000100) CLOSE(0x80000000)	1sttest.txt	5292	30
②	19734691272	2017-11-27 11:26:37	FILE_DELETE(0x00000200) CLOSE(0x80000000)	1sttest.txt	5292	30
③	19734692320	2017-11-27 11:26:52	FILE_CREATE(0x00000100)	2ndtest.txt	5292	31
	19734692408	2017-11-27 11:26:52	DATA_EXTEND(0x00000002) FILE_CREATE(0x00000100)	2ndtest.txt	5292	31
	19734694200	2017-11-27 11:27:08	DATA_EXTEND(0x00000002) FILE_CREATE(0x00000100) CLOSE(0x80000000)	2ndtest.txt	5292	31

[그림3-13] MFT Entry 번호 5292의 $UsnJrnl 레코드 이력

제 4 장 $UsnJrnl 파일의 수집 근거 및 필요성

1. $UsnJrnl 파일의 증거로서의 가치와 증거능력

디지털 증거에 대한 개념정의의 예를 살펴보면, '범죄를 입증하거나 범죄와 피해자 또는 범죄자 사이의 연결 고리를 제공할 수 있는 모든 디지털 데이터(Eoghan Casey)', '범죄가 어떻게 일어났는지에 대하여 입증 또는 반박할 수 있거나, 범죄 의도나 알리바이와 같은 범죄의 핵심 요소들을 이끌어 낼 수 있는 정보로 컴퓨터를 사용하여 저장되거나 전송되는

데이터(W. Jerry Chrisum)'라고 정의되기도 하고,[23] 또 '디지털 형태로 저장 또는 전송되는 증거가치 있는 정보(SWGDE[24])', '법정에서 신뢰할 수 있는 저장되거나 전송되는 이진수 형태의 정보(IOCE[25])', '각종 디지털 저장매체에 저장되거나 네트워크 장비 및 유·무선 통신상으로 전송되는 정보 중 그 신뢰성을 보장할 수 있어 증거로서 가치를 가지는 디지털 정보'로 정의되기도 한다.[26]

이러한 디지털 증거가 효력을 갖기 위해서는 적법절차를 통해 수집되고, 데이터를 처리하는데 있어 그 결과는 항상 동일한 결과를 산출해야 하며, 절차의 연속성이 확보되어야 하고, 그 정보가 무결함이 입증되어야 한다.[27] 이는 디지털 정보가 가지는 변개의 용이성, 취약성, 익명성 등의 특징들로 인한 것으로 위의 요건들이 만족되면 증거능력을 인정하고 있다.

사용자의 행위나 시스템의 모든 실행 내역 등을 기록하는 $UsnJrnl 파일의 기능적, 내용적 특성들은 범죄 사실과의 연관성을 입증하는데 직접적 혹은 간접적인 방법으로 활용될 수 있으며 사용자에 의한 변조가 불가능한 시스템 영역의 데이터라는 객관성과 신뢰성을 가진다. 따라서 수집 과정에서의 절차적 정당성과 무결성 등 기본 원칙이 확보되면 증거능력을 인정받을 수 있다.

---

23) Eoghan Casey, 「Digital Evidence and Comuter Crime(2nd ed.).」 Academic Press(2004), 12

24) Scientific Working Group on Digital Evidence : 미국 법무부의 마약수사청, 연방수사 관세청 및 국세청 등의 증거분석 연구소를 중심으로 구성된 디지털 증거에 관한 과학실무그룹

25) International Orgarnisation on Computer Evidence : 미국, 호주, 홍콩, 영국 등 각국의 실무자들을 중심으로 만들어진 디지털 증거에 관한 국제기구

26) 탁희성, 이상진, "디지털 증거분석 도구에 의한 증거수집절차 및 증거능력 확보방안" 한국형사정책연구원(2006), 31-34

27) 손지영, 김주석, "디지털 증거의 증거능력 판단에 관한 연구" 사법정책연구원 연구총서 2015-08, 대법원 사법정책연구원(2015), 44

2. $UsnJrnl 파일의 사건 관련성 여부 판단

디지털 정보를 포함한 모든 증거들은 사건과의 관련성 유무를 통해 그 수집의 가능 여부를 판단한다. 우리 형사소송법은 제215조를 통해 '해당 사건과 관계가 있다고 인정할 수 있는 것'에 한하여 압수수색이 가능하다고 규정하고 있고, 제106조 제3항을 통해 '압수의 목적물이 컴퓨터용 디스크, 그 밖에 이와 비슷한 정보저장매체인 경우에는 기억된 정보의 범위를 정하여 출력하거나 복제하여 제출받아야 한다'는 관련성 있는 정보에 대한 선별 압수의 원칙을 규정하고 있다. 또한 실무 규정인 대검찰청 예규 제876호 「디지털 증거의 수집·분석 및 관리 규정(개정 2016.12.26.)」 제15조 제3항에서도 '주임검사 주도 하에 피압수자 등을 참여시킨 상태에서 사건과 관련성이 있는 정보를 선별하도록 규정'하고 있다.

$UsnJrnl 파일의 특성상 이 파일은 시스템 영역과 사용자 영역 모두를 기록한다. 이 파일에는 사용자의 모든 행위가 기록되기는 하나 변경 여부만 기록될 뿐 직접적인 데이터 정보를 가지지는 않는다. 즉, 사건과 무관한 데이터를 기록한다고 해도 해당 파일만으로는 그 내용을 구체적으로는 알 수 없다.

이 파일은 선별을 위한 탐지 과정에서 사용되는 파일로서의 기능적 측면이 크고, $UsnJrnl 파일로부터 탐지된 정보를 통해 2차적으로 접근한 파일에 대한 보강적 기능이 크다. 따라서 위 파일에 대한 관련성 범위는 포괄적으로 인정할 필요가 있다.

3. $UsnJrnl 파일의 전문법칙 적용 여부

디지털 증거는 사람에 의한 처리과정의 유무에 따라 컴퓨터에 의하여 생성된 증거, 컴퓨터의 조력을 받은 증거로 구분할 수 있다. 전자는 전문법칙의 적용을 받지 않는 반면 후자는 전문법칙이 적용될 수 있다는 점

에서 차이가 있다.[28] 전문법칙이란 전해들은 증거, 즉 전문 증거는 증거로 되지 않는다는 법원칙을 말하는데 이는 원진술자의 반대신문권을 보장하기 위함이다. 따라서 전문 증거의 경우에는 원진술자가 공판정에서 진정의 성립을 인정하는 때에 증거로써의 능력을 갖는다. 반면 비진술 증거는 전문법칙의 예외에 해당하는데 통상문서, 기계적으로 작성한 장부 등이 이에 해당하고 이러한 비진술 증거는 진정성이 인정되는 경우 증거 능력을 부여 받는다.

증거능력과 관련된 해외 입법례를 살펴보면 미국 연방증거법 제901조(b)는 증거능력을 인정받기 위한 전제조건으로 진정성과 동일성의 입증요건 10가지를 예시하고 있는데 그 중 제9호 절차와 시스템은 '어떤 결과물을 산출하는데 이용하는 절차나 시스템을 설명하는 증거와 그 절차, 시스템이 정확한 결과를 산출한다는 것을 보여주는 증거'를 말한다.[29] 독일에서는 '타인의 조사 내용으로 법원의 조사 대체를 금지하는 직접주의가 존재한다. 따라서 제3자가 타인의 진술을 기재한 서면은 원진술자를 법정에 소환하여 신문하지 않는 이상 증거사용이 불가능하다. 하지만 직접주의가 적용되는 서면은 처음부터 증거로 제출될 목적으로 작성된 것을 말한다고 한다. 따라서 증거목적으로 작성되지 않은 문건은 직접주의에 의한 금지대상이 아니어서 그 자체를 증거로 할 수 있다' 라고 규정하면서 전문법칙의 예외를 광범위하게 인정하고 있다.

$UsnJrnl 파일은 사용자에 의해 생성되는 정보가 아니다. 사용자가 해당 파일에 대한 활성화, 비활성화의 여부만 결정할 수 있을 뿐 그 내용은 사용자의 의도대로 기록되지 않고 내용에 대한 변조가 불가능한 역역으로 보인다. 따라서 컴퓨터에 의해 생성된 정보로서 전문법칙의 적용을 받지 않는 것으로 판단된다.

---

28) 김영기, "디지털 증거의 진정성립부인과 증거능력 부여 방안", 한국형사판례연구(19), 2011, 514

29) Evidence describing a process or system used to produce a result and showing that the process or system produces an accurate result.

4. $UsnJrnl 파일의 수집 필요성 및 방법

현재 우리나라의 경우 현장에서의 사건 관련성 있는 파일들에 대한 선별 압수를 원칙으로 하고 있어 $UsnJrnl 파일과 같은 시스템에서 생성하는 파일들이나 비할당영역에서의 복구 가능성이 있는 파일들에 대한 확보가 어렵다. 일반 범죄에서 디지털 정보기기가 관련되어 있는 경우 해당 기기에서 혐의에 대한 단서 또는 직접적인 증거를 확보하거나, 정황을 좀 더 구체화하기 위한 보조자료 등을 얻을 수도 있다. 보편적으로 얻을 수 있는 정보는 대략 언제, 어떤 프로그램이 수행되었는지에 관한 정보와 문서를 작성한 경우 그 파일내용 등이다. 또한 인터넷을 통하여 웹사이트를 방문한 경우에는 방문한 웹사이트 주소, 접속 시간 등과 같이 의외로 많은 정보를 얻을 수 있다.[30] 따라서 이러한 정보들을 기록하는 시스템이 생성하는 정보들은 그 수집의 필요성을 폭넓게 인정할 필요가 있다.

범행현장성에 따른 디지털 포렌식 절차 모델 개발 연구(이승무, 2016)에서는 디지털 증거에 대한 압수수색 방법을 결정하기 위한 3가지 결정요소, 이를테면 '범행현장성', '선별압수 가능성', '개괄적 선별조치 가능성'을 제시하면서 압수수색 대상 증거가 범행의 직접적인 대상이 되었거나 범행의 필요적 도구로써 사용되었거나, 기타 정황상 범행과 불가분의 관계에 있음이 명백한 경우, 해당 디지털 증거는 '범행현장성'이 있는 증거로 간주하고, 이러한 경우에는 원칙적으로 원본을 압수할 것을 제안했다.[31]

만약 범행현장성의 요건을 갖추지 못하는 등의 원본 압수가 불가능한 경우라면 현장에서 해당 파일을 확보할 때 현장에서의 개별 파일 수집의 필요성을 우선 판단해야 할 것인데, 특히 현장에서의 $UsnJrnl 파일에 대한 수집 여부 판단 시 고려해야 할 사안이라면, 쟁점이 되는 것이 무엇인지에 대한 판단일 것인 바, 만약 사용자의 전반적인 행위 중 혐의와

30) 손지영 외 1, 앞의 글, 48-49
31) 이승무, "범행현장성에 따른 디지털 포렌식 절차 모델 개발" 서울대 융합과학기술대학원 수리정보과학과 디지털포렌식학 석사 학위 논문(2017), 초록 외

관련된 특정 행위를 탐지해야 할 때, 혹은 특정 파일에 대한 생성, 삭제, 변경 등의 행위와 그 시간을 탐지해야 할 때라면 위 파일은 반드시 확보할 필요가 있을 것으로 보인다.

제 5 장 실무에서의 $UsnJrnl 파일 활용 사례

이번 장에서는 실제 검찰청의 디지털포렌식 분석 사건 중 $UsnJrnl을 활용한 사례가 있어 어떠한 방법으로 활용이 되었는지 살펴보고자 한다. 아래에서 살펴볼 사례들은 원본 압수된 기기들을 대상으로 분석한 사안들이고, 해당 내용은 디지털포렌식 수사관이 작성한 분석보고서만을 활용하였다.

1. 기술유출 사례

해당 사건은 퇴사 직전 피해 회사의 핵심기술 설계도면, 영업실적 등 영업 비밀을 유출하고 동종의 업체에 입사하여 본 기술을 사용한 산업기술의유출방지및보호에관한법률위반 사건으로, 수사팀으로부터 기술유출과 관련된 일체의 정보에 대한 분석을 요청 받은 사례다. 피의자는 기술유출의 의심을 피하고자 본인이 사용하던 하드디스크에 존재하는 해당 기술을 포함한 파일들을 모두 삭제한 것으로 추정된다.

분석 과정에서 담당 분석관은 $UsnJrnl 파일에서 유출된 기술명과 유사한 이름의 파일 세 개가 2015. 1. 20. 16:36:31에 일괄 삭제된 흔적을 발견하였는데 이를 통해 삭제 날짜가 피의자의 퇴사 일자가 동일하다는 것을 확인할 수 있었다.

일반적으로 파일을 삭제하는 경우 삭제된 원본 파일이 존재하지 않기 때문에 그 삭제시기를 특정하기 어렵다. 만약 파일 카빙을 통해 원본 파일이 발견된다 하더라도 그 파일이 MFT 영역에 의해 복구된 파일이 아

니므로 그 직접적인 연관성 또한 불확실하다. 일반적으로 분석관들이 파일 삭제 시간을 추정하는 방법은 해당 파일이 존재했던 파일의 상위 디렉토리의 MFT Entry 변경 시간을 통해 추정하거나 레지스트리에 남아있는 최근 실행된 이력을 통해 잔존했던 최후의 시간으로부터 추정한다. 그러나 이러한 방법 또한 원본 파일이 존재하던 위치와 동일한 위치에 다른 파일들도 함께 존재하는 경우 구체성이 떨어지고 해당 파일을 실행하기 않고 삭제하는 경우 최근 이력을 통해 잔존했던 최후 시기에 대한 특정은 불확실하다.

더 나아가 발견된 레코드들을 통해 그 삭제 방법도 좀 더 구체화 할 수 있다. 일반적으로 사용자가 직접 특정 파일을 지정하고 이를 삭제하는 경우 File_Renamed_Old>File_Renamed_New>File_Renamed_New/File_Closed의 과정을 통해 삭제행위가 이루어진다. 하지만 해당 분석보고서에 첨부된 화면 캡쳐 그림을 보면 해당 파일들이 File_Closed > File_Deleted 의 과정으로 일괄 삭제가 된 것을 볼 수 있다. 이러한 일괄삭제 행위는 보통 여러 개의 파일을 모두 지정한 후 한꺼번에 삭제하는 경우나, 특정 프로그램에 의해 삭제되는 경우 발생한다. 이러한 흔적이 발견될 경우 이전 레코드들을 확인하여 삭제 프로그램의 사용 여부 등을 추가로 조사해 볼 수 있겠다.

<표15> 분석 화면 캡쳐 (분석도구 : XWF)

[그림5-1] $UsnJnrl 파일 내 특정 파일이 삭제된 레코드 정보

2. 연예인 얼굴을 합성한 음란물 유포 사례

해당 사건은 P2P 프로그램을 통해 연예인 얼굴을 합성한 음란물 사진을 다운로드 받아 이를 유포한 사건이다. 검찰 조사 직전에 관련 이미지 등 데이터들을 삭제한 것으로 추정되어, 수사팀으로부터 삭제된 합성 사

진에 대한 복구 요청과 함께 삭제시기에 대한 특정을 요청받은 사례이다.

위 사건의 경우 할당영역의 $UsnJrnl 파일에는 약 이틀가량의 기록만이 남아 있었다. 피압수자의 $UsnJrnl 파일을 확인해 보면 피의자의 컴퓨터 사용 습성 등의 행위 추적이 가능한데, 위의 경우 당 사건 피의자는 대부분 인터넷을 통해 게임 프로그램 등을 다운로드 받아 사용하였고 지속적으로 시스템을 켜둔 상태로 생활하는 것을 알 수 있었다. 비할당영역을 대상으로 합성 사진 원본 이미지 복구를 시도하였으나 혐의와 관련된 파일은 발견되지 않았고, 유일한 단서는 ThumbnailCache[32] 파일이었으나 이 Thumbnail 이미지로는 원본 파일 이름을 확인할 수 없다. 원본 파일 이름이 확인되지 않으면 그 삭제 시기 또한 추정하기가 어렵다. 하지만 위의 사진들이 저장되어 있던 것으로 추정되는 '합성사진!.zip' 파일을 발견하였고, 원본 사진 파일명이 특정되지 않은 상태에서 위 '합성사진!.zip' 파일의 삭제시기가 중요 쟁점으로 판단되어 해당 파일에 대한 분석을 실시하였고, 해당 파일이 다른 파일에 의해 덮어쓰여진 사실을 $UsnJrnl 파일에서 확인할 수 있었다.

아래 그림과 같이 할당영역의 $UsnJrnl 파일에서 '합성사진!zip'는 2017.09.12 15:31:47에 'Ass~'라는 게임 프로그램에 의해 덮어써진 것을 확인할 수 있다.

[그림5-2] 합성사진.zip 파일이 다른 파일로 덮어쓰여진 레코드 정보

32) 윈도우 탐색기의 미리보기 기능에 사용되는 파일로 원본 이미지에 대한 축소 이미지를 저장한다. 이것은 원본 이미지가 해당 기기에 저장되어 있는 것을 의미하며, 사용자가 윈도우 탐색기를 통해 한번이라도 미리보기를 수행했다면 thumbnailcache 파일로 저장된다.

덮어쓰여진 시간이 삭제시간을 의미하는 것은 아니나, 해당파일이 존재했던 위치 정보나 파일명을 확인할 수 있고, '합성사진!.zip' 파일이 마지막으로 실행된 흔적을 추가적으로 분석한다면 위 파일이 삭제된 근접한 시간대를 특정해 볼 수 있을 것이다.

$UsnJrnl 파일은 다른 파일들과 달리 사용자의 행위를 포함한 시스템의 정확한 이벤트 시간을 알 수 있다는데 큰 의미가 있다. 특히 삭제 시기를 다투는 경우, 보통 상위 디렉토리의 MFT 수정시간을 통해 삭제시간을 추측하지만 $UsnJrnl에서 확인되는 경우 실제 행위 시간을 정확히 파악할 수 있음을 사례를 통해 확인할 수 있었다.

제 6 장 가상 시나리오를 통한 활용 방안 제안

$UsnJrnl 파일을 중점으로 사건을 분석해 보고자 증거인멸과 기술유출 사건의 가상 시나리오를 설정하고 환경을 구축하여 실험을 진행하였다. 두 가지 사례는 실제 사건을 일부 참고하여 설정하였으며 시간정보를 기반으로 한 사용자의 행위 추적에 그 분석 초점을 두었다. 두 사례는 각각 Windows 7과 Windows 8 두 가지 다른 버전을 사용하였으며 해당 사례를 분석하는 도구들은 실제 검찰청에서 사용하는 디지털 포렌식 도구들과 인터넷을 통해 다운로드 받아 사용 가능한 소프트웨어를 사용하였다.

각 사례들에 대한 분석 방법은 본 연구의 취지에 맞게 $UsnJrnl 파일에서 발견된 단서를 시작으로 상세 분석을 진행할 것이다. 이는 $UsnJrnl 파일이 하나의 단서로 사용될 수 있음을 보여주기 위함이다.

구체적으로 '(1) $UsnJrnl 파일을 통한 사용자 행위 추적'에서는 전반적인 사용자나 시스템의 변화에 대해 간략히 해석하고, 이어지는 '(2)단서로서의 $UsnJrnl 파일 활용'에서 위에서 탐지된 단서를 통해 구체적인 분석을 이어가도록 하겠다.

1. 증거인멸 사례

시 나 리 오
피고인은 디지털증거분석 업무를 맡고 있는 수사기관 직원이다. 피고인은 현재 K사건으로 검찰 조사를 받고 있는 前팀장으로부터 그의 담당 업무와 관련된 문서와 데스크톱 PC를 인계 받았다. 피고인은 위의 인계받은 문건들을 검토하던 중 前팀장이 관리하던 문서들 중에서 K사건과 관련된 증거들 중 일부에 문제가 있음을 발견하고 이후 관련 사건으로 자신도 조사를 받게 될 것을 우려하여 해당 문건을 삭제하기로 마음먹고 다음과 같이 증거인멸을 시도하였다. - 관련 문건 삭제 - 사건 발생 이전의 시간으로 시스템 시간 변경 - 변경된 시간에서 특정 디렉토리와 디렉토리 내 파일 모두 삭제 - 이벤트 로그 삭제 - 디스크 조각 모음 실행 - 현재의 시간으로 시스템 시간 재변경 - 완전 삭제 프로그램 'Moo0 Anti Recovery 1.11' 다운로드 - 'Moo0 Anti Recovery 1.11'로 파일 제목 완전 삭제 실행 - 'Moo0 Anti Recovery 1.11' 프로그램 삭제

(1) 분석 환경 및 분석 방법

① 가상 시나리오 시스템 정보 및 사용 도구

대상	내용
파일명	증거인멸사례.E01
운영체제	Windows 7 Professional
파일시스템	NTFS
표준시간대	UTC +09:00
증거사본작성 및 파일 추출, 분석 도구	EnCase v7.12.01
$UsnJrnl 분석 도구	NTFS Log Tracker, NTFS $UsnJrnl Parser(v5.0.1)[33]
프리패치 분석 도구	WinPrefetchView v1.3.5
이벤트로그 분석 도구	Event Log Explorer v4.6.1.2115

33) Guidance Software에서 EnCase 에서의 작업을 자동화하고, 기능을 활용하기 위해 설계된 언어

② 가상 시나리오 타임라인 (시스템 시간 기준)

시간	행위 → 방법
2017.11.11. 18:30 경	5개의 대상 문건 일괄 삭제 → Delete 키 이용
2017.11.11. 18:40 경	시스템 시간 변경 → 윈도우 시스템의 시간 변경 기능 이용
2017.10.11. 18:41 경	3개의 문건이 들어있는 1개의 디렉토리 삭제 → Delete 키 이용
2017.10.11. 18:42 경	이벤트로그 삭제 및 디스크조각모음 실행 → 윈도우 시스템으로 삭제 및 실행
2017.10.11. 18:50 경	시스템 시간 변경 → 변경 방법 : 인터넷 시간>지금업데이트
2017.11.11. 19:05 경	삭제프로그램 'Moo0 Anti Recovery 1.11' 다운로드 → 미실행
2017.11.11. 19:35 경	'Moo0 Anti Recovery 1.11' 실행 → 5번 파일이름흔적 선택
2017.11.11. 19:45 경	'Moo0 Anti Recovery 1.11' 삭제 → 프로그램 실행 및 삭제

(2) 분석 결과 및 활용 방법

(1) $UsnJrnl 파일을 통한 사용자 행위 추적

아래의 그림은 $UsnJrnl 파일의 레코드들 중 혐의와 관련 있어 보이는 부분을 시간의 순서대로 나열한 것이다. 이 레코드들을 통해서 구체적인 행위 시간 정보, 실행되는 패치파일 정보, 접근 경로 등을 확인해 볼 수 있다.

시간 / 행위

레코드

① 2017.11.11. 18:30:20 / 5개의 대상 문건 일괄 삭제

USN	레코드 시간 정보	변경 원인 플래그	대상 파일 이름	대상 파일 타입	대상 파일의 MFT Entry 번호	대상 파일의 MFT Entry Sequence 번호	전체 경로
26337168	2017-11-11 18:30:20	FILE_DELETE(0x00000200) CLOSE(0x80000000)	보안.patch	Archive (0x00000020)	64834	6	D₩Users₩Yun_Desktop₩Desktop
26337280	2017-11-11 18:30:20	FILE_DELETE(0x00000200) CLOSE(0x80000000)	증거분석결과보고서.pdf	Archive (0x00000020)	64183	9	D₩Users₩Yun_Desktop₩Desktop
26337368	2017-11-11 18:30:20	FILE_DELETE(0x00000200) CLOSE(0x80000000)	K_사건관련 중간보고.pptx	Archive (0x00000020)	64845	1	D₩Users₩Yun_Desktop₩Desktop
26337464	2017-11-11 18:30:20	FILE_DELETE(0x00000200) CLOSE(0x80000000)	보도자료_스크랩.JPG	Archive (0x00000020)	64846	1	D₩Users₩Yun_Desktop₩Desktop
26337552	2017-11-11 18:30:20	FILE_DELETE(0x00000200) CLOSE(0x80000000)	K_사건관련보고사항정리.hwp	Archive (0x00000020)	64847	1	D₩Users₩Yun_Desktop₩Desktop

② 2017.11.11. 18:40:29 / 시스템 시간 변경 (2017.11.11.에서 2017.10.11.로 변경)

USN	레코드 시간 정보	변경 원인 플래그	대상 파일 이름	대상 파일 타입	대상 파일의 MFT Entry 번호	대상 파일의 MFT Entry Sequence 번호	전체 경로
26342160	2017-11-11 18:40:29	DATA_EXTEND(0x00000002) DATA_TRUNCATION(0x00000004) CLOSE(0x80000000)	RUNDLL32.EXE-89545801.pf	Archive (0x00000020), Not Content Indexed	65383	8	D₩Windows₩Prefetch
26342272	2017-10-11 18:40:31	DATA_TRUNCATION(0x00000004)	DLLHOST.EXE-C373C89E.pf	Archive (0x00000020), Not Content Indexed	65112	1	D₩Windows₩Prefetch

③ 2017.10.11. 18:41:06 / 3개의 문건이 들어있는 1개의 디렉토리 삭제

USN	레코드 시간 정보	변경 원인 플래그	대상 파일 이름	대상 파일 타입	대상 파일의 MFT Entry 번호	대상 파일의 MFT Entry Sequence 번호	전체 경로
26342936	2017-10-11 18:41:06	RENAME_OLD_NAME (0x00001000)	K_사건관련각종보고		23558	25	D₩Users₩Yun_Desktop₩Desktop
26343016	2017-10-11 18:41:06	RENAME_NEW_NAME (0x00002000)	$R09J51J		23558	25	D₩$Recycle.Bin₩S-1-5-21-2025115636-2423932730-2888420290-1001
26343096	2017-10-11 18:41:06	RENAME_NEW_NAME (0x00002000) CLOSE(0x80000000)	$R09J51J		23558	25	D₩$Recycle.Bin₩S-1-5-21-2025115636-2423932730-2888420290-1001
26343176	2017-10-11 18:41:06	SECURITY_CHANGE (0x00000800)	월간보고.hwp	Archive (0x00000020)	64808	2	D₩$Recycle.Bin₩S-1-5-21-2025115636-2423932730-2888420290-1001₩K_사건관련각종보고
26343256	2017-10-11 18:41:06	SECURITY_CHANGE(0x00000800) CLOSE(0x80000000)	월간보고.hwp	Archive (0x00000020)	64808	2	D₩$Recycle.Bin₩S-1-5-21-2025115636-2423932730-2888420290-1001₩K_사건관련각종보고

26343336	2017-10-11 18:41:06	SECURITY_CHANGE(0x00000800)	일일업무보고.hwp	Archive (0x00000020)	64804	2	D₩$Recycle.Bin₩S-1-5-21-2025115636-2423932730-2888420290-1001₩K_사건관련각종보고
26343416	2017-10-11 18:41:06	SECURITY_CHANGE(0x00000800) CLOSE(0x80000000)	일일업무보고.hwp	Archive (0x00000020)	64804	2	D₩$Recycle.Bin₩S-1-5-21-2025115636-2423932730-2888420290-1001₩K_사건관련각종보고
26343496	2017-10-11 18:41:06	SECURITY_CHANGE(0x00000800)	주간보고.hwp	Archive (0x00000020)	64805	2	D₩$Recycle.Bin₩S-1-5-21-2025115636-2423932730-2888420290-1001₩K_사건관련각종보고
26343576	2017-10-11 18:41:06	SECURITY_CHANGE(0x00000800) CLOSE(0x80000000)	주간보고.hwp	Archive (0x00000020)	64805	2	D₩$Recycle.Bin₩S-1-5-21-2025115636-2423932730-2888420290-1001₩K_사건관련각종보고
26343656	2017-10-11 18:41:06	SECURITY_CHANGE(0x00000800)	$R09J51J		23558	25	D₩$Recycle.Bin₩S-1-5-21-2025115636-2423932730-2888420290-1001
26343736	2017-10-11 18:41:06	SECURITY_CHANGE(0x00000800)CLOSE(0x80000000)	$R09J51J		23558	25	D₩$Recycle.Bin₩S-1-5-21-2025115636-2423932730-2888420290-1001

④ 2017.10.11. 18:42:26~18:43:16 / 이벤트로그 삭제 및 디스크조각모음 실행

USN	레코드 시간 정보	변경 원인 플래그	대상 파일 이름	대상 파일 타입	대상 파일의 MFT Entry 번호	대상 파일의 MFT Entry Sequence 번호	전체 경로
26344392	2017-10-11 18:42:26	DATA_OVERWRITE(0x00000001)	eventvwr	Archive (0x00000020), Not Content Indexed	65109	2	D₩Users₩Yun_Desktop₩AppData₩Roaming₩Microsoft₩MMC
26344472	2017-10-11 18:42:26	DATA_OVERWRITE(0x00000001) CLOSE(0x80000000)	eventvwr	Archive (0x00000020), Not Content Indexed	65109	2	D₩Users₩Yun_Desktop₩AppData₩Roaming₩Microsoft₩MMC
26344552	2017-10-11 18:42:26	DATA_TRUNCATION(0x00000004)	RecentViews	Archive (0x00000020), Not Content Indexed	65110	1	D₩Users₩Yun_Desktop₩AppData₩Local₩Microsoft₩Event Viewer
26344640	2017-10-11 18:42:26	DATA_EXTEND(0x00000002) DATA_TRUNCATION(0x00000004)	RecentViews	Archive (0x00000020), Not Content Indexed	65110	1	D₩Users₩Yun_Desktop₩AppData₩Local₩Microsoft₩Event Viewer
26344728	2017-10-11 18:42:26	DATA_EXTEND(0x00000002) DATA_TRUNCATION(0x00000004) CLOSE(0x80000000)	RecentViews	Archive (0x00000020), Not Content Indexed	65110	1	D₩Users₩Yun_Desktop₩AppData₩Local₩Microsoft₩Event Viewer

USN	레코드 시간 정보	변경 원인 플래그	대상 파일 이름	대상 파일 타입	대상 파일의 MFT Entry 번호	대상 파일의 MFT Entry Sequence 번호	전체 경로
26345344	2017-10-11 18:43:16	DATA_TRUNCATION(0x00000004)	DFRGUI.EXE-C853DD35.pf	Archive (0x00000020), Not Content Indexed	64545	2	D₩Windows₩Prefetch
26345472	2017-10-11 18:43:16	DATA_EXTEND(0x00000002) DATA_TRUNCATION(0x00000004)	DFRGUI.EXE-C853DD35.pf	Archive (0x00000020), Not Content Indexed	64545	2	D₩Windows₩Prefetch
26345576	2017-10-11 18:43:16	DATA_EXTEND(0x00000002) DATA_TRUNCATION(0x00000004) CLOSE(0x80000000)	DFRGUI.EXE-C853DD35.pf	Archive (0x00000020), Not Content Indexed	64545	2	D₩Windows₩Prefetch

⑤ 2017.10.11. 18:43:26 / 시스템 시간 변경(2017.10.11.에서 2017.11.11.로 원위치)

USN	레코드 시간 정보	변경 원인 플래그	대상 파일 이름	대상 파일 타입	대상 파일의 MFT Entry 번호	대상 파일의 MFT Entry Sequence 번호	전체 경로
26346208	2017-10-11 18:43:26	DATA_EXTEND(0x00000002) DATA_TRUNCATION(0x00000004) CLOSE(0x80000000)	DFRGUI.EXE-C853DD35.pf	Archive (0x00000020), Not Content Indexed (0x00002000)	64545	2	D₩Windows₩Prefetch
26346312	2017-11-11 18:50:59	SECURITY_CHANGE(0x00000800)	76516719-0781-429f-8363-196b086d6adb.png	Archive (0x00000020), Not Content Indexed (0x00002000)	63579	6	D₩Windows₩ServiceProfiles₩NetworkService₩AppData₩Local₩Microsoft₩Windows Media Player NSS₩3.0₩Icon Files
26346456	2017-11-11 18:50:59	SECURITY_CHANGE(0x00000800) CLOSE(0x80000000)	76516719-0781-429f-8363-196b086d6adb.png	Archive (0x00000020), Not Content Indexed (0x00002000)	63579	6	D₩Windows₩ServiceProfiles₩NetworkService₩AppData₩Local₩Microsoft₩Windows Media Player NSS₩3.0₩Icon Files
26346600	2017-11-11 18:50:59	SECURITY_CHANGE(0x00000800)	SCPD	Not Content Indexed (0x00002000)	23285	2	D₩Windows₩ServiceProfiles₩NetworkService₩AppData₩Local₩Microsoft₩Windows Media Player NSS₩3.0
26346672	2017-11-11 18:50:59	SECURITY_CHANGE(0x00000800) CLOSE(0x80000000)	SCPD	Not Content Indexed (0x00002000)	23285	2	D₩Windows₩ServiceProfiles₩NetworkService₩AppData₩Local₩Microsoft₩Windows Media Player NSS₩3.0

⑥ 2017.11.11. 19:05:13 / 삭제프로그램 'Moo0 Anti Recovery 1.11' 다운로드

USN	레코드 시간 정보	변경 원인 플래그	대상 파일 이름	대상 파일 타입	대상 파일의 MFT Entry 번호	대상 파일의 MFT Entry Sequence 번호	전체 경로
26528880	2017-11-11 19:05:14	FILE_CREATE(0x00000100)	Moo0_Anti_Recovery_v1.11_Installer.exe	Archive (0x00000020)	23081	8	D₩Users₩Yun_Desktop₩Downloads
26529016	2017-11-11 19:05:14	DATA_EXTEND(0x00000002) FILE_CREATE(0x00000100)	Moo0_Anti_Recovery_v1.11_Installer.exe	Archive (0x00000020)	23081	8	D₩Users₩Yun_Desktop₩Downloads
26529152	2017-11-11 19:05:14	DATA_OVERWRITE(0x00000001) DATA_EXTEND(0x00000002) FILE_CREATE(0x00000100)	Moo0_Anti_Recovery_v1.11_Installer.exe	Archive (0x00000020)	23081	8	D₩Users₩Yun_Desktop₩Downloads
26529288	2017-11-11 19:05:14	DATA_OVERWRITE(0x00000001) DATA_EXTEND(0x00000002) FILE_CREATE(0x00000100) BASIC_INFO_CHANGE(0x00008000)	Moo0_Anti_Recovery_v1.11_Installer.exe	Archive (0x00000020)	23081	8	D₩Users₩Yun_Desktop₩Downloads
26529424	2017-11-11 19:05:14	DATA_OVERWRITE(0x00000001) DATA_EXTEND(0x00000002) FILE_CREATE(0x00000100) BASIC_INFO_CHANGE(0x00008000) CLOSE(0x80000000)	Moo0_Anti_Recovery_v1.11_Installer.exe	Archive (0x00000020)	23081	8	D₩Users₩Yun_Desktop₩Downloads

~중략~

26537112	2017-11-11 19:05:51	FILE_CREATE(0x00000100)	Moo0		23642	9	D₩Program Files (x86)
26537184	2017-11-11 19:05:51	FILE_CREATE(0x00000100) CLOSE(0x80000000)	Moo0		23642	9	D₩Program Files (x86)

~중략~

26537256	2017-11-11 19:05:51	FILE_CREATE(0x00000100)	Anti-Recovery 1.11		23675	13	D₩Program Files (x86)₩Moo0
26537352	2017-11-11 19:05:51	FILE_CREATE(0x00000100) CLOSE(0x80000000)	Anti-Recovery 1.11		23675	13	D₩Program Files (x86)₩Moo0

~중략~

26539864	2017-11-11 19:05:51	FILE_CREATE(0x00000100)	Moo0 데이타 회복방지기 1.11.lnk	Archive (0x00000020)	23704	8	D₩Users₩Yun_Desktop₩Desktop
26539976	2017-11-11 19:05:51	DATA_EXTEND(0x00000002) FILE_CREATE(0x00000100)	Moo0 데이타 회복방지기 1.11.lnk	Archive (0x00000020)	23704	8	D₩Users₩Yun_Desktop₩Desktop
26540088	2017-11-11 19:05:51	DATA_EXTEND(0x00000002) FILE_CREATE(0x00000100) CLOSE(0x80000000)	Moo0 데이타 회복방지기 1.11.lnk	Archive (0x00000020)	23704	8	D₩Users₩Yun_Desktop₩Desktop

⑦ 2017.11.11. 19:35:16~ / 'Moo0 Anti Recovery 1.11' 실행

USN	레코드 시간 정보	변경 원인 플래그	대상 파일 이름	대상 파일 타입	대상 파일의 MFT Entry 번호	대상 파일의 MFT Entry Sequence 번호	전체 경로
26593464	2017-11-11 19:35:16	FILE_CREATE(0x00000100)	ANTIRECOVERY.EXE-AF101B70.pf	Archive (0x00000020), Not Content Indexed	23991	9	D₩Windows₩Prefetch
26593584	2017-11-11 19:35:16	DATA_EXTEND(0x00000002) FILE_CREATE(0x00000100)	ANTIRECOVERY.EXE-AF101B70.pf	Archive (0x00000020), Not Content Indexed	23991	9	D₩Windows₩Prefetch
26593704	2017-11-11 19:35:16	DATA_EXTEND(0x00000002) FILE_CREATE(0x00000100) CLOSE(0x80000000)	ANTIRECOVERY.EXE-AF101B70.pf	Archive (0x00000020), Not Content Indexed	23991	9	D₩Windows₩Prefetch
26593824	2017-11-11 19:35:22	FILE_CREATE(0x00000100)	Moo0_AntiRecovery.TMP.000000		24010	6	₩moo0_AntiRecovery path of parent entry
26593944	2017-11-11 19:35:22	FILE_CREATE(0x00000100) CLOSE(0x80000000)	Moo0_AntiRecovery.TMP.000000		24010	6	Unable to resolve path of parent entry
26594064	2017-11-11 19:35:22	FILE_CREATE(0x00000100)	0	Archive (0x00000020)	24022	5	Unable to resolve path of parent entry
26594576	2017-11-11 19:35:22	FILE_CREATE(0x00000100) CLOSE(0x80000000)	0	Archive (0x00000020)	24022	5	Unable to resolve path of parent entry
26595328	2017-11-11 19:35:22	FILE_CREATE(0x00000100)	1	Archive (0x00000020)	24031	3	Unable to resolve path of parent entry
26595840	2017-11-11 19:35:22	FILE_CREATE(0x00000100) CLOSE(0x80000000)	1	Archive (0x00000020)	24031	3	Unable to resolve path of parent entry

~중략~

26603520	2017-11-11 19:35:22	FILE_CREATE(0x00000100)	9	Archive (0x00000020)	24152	3	Unable to resolve path of parent entry
26604032	2017-11-11 19:35:22	FILE_CREATE(0x00000100) CLOSE(0x80000000)	9	Archive (0x00000020)	24152	3	Unable to resolve path of parent entry
26604544	2017-11-11 19:35:22	FILE_CREATE(0x00000100)	000000000000000000000000 000000000000000000000000 000000000000000000000000 000000000000000000000000	Archive (0x00000020)	63609	15	Unable to resolve path of parent entry
26605056	2017-11-11 19:35:22	FILE_CREATE(0x00000100) CLOSE(0x80000000)	000000000000000000000000 000000000000000000000000 000000000000000000000000 000000000000000000000000	Archive (0x00000020)	63609	15	Unable to resolve path of parent entry
			~중략~				
26927616	2017-11-11 19:35:23	RENAME_OLD_NAME(0x00001000)	0	Not Content Indexed (0x00002000)	24022	5	Unable to resolve path of parent entry
26928128	2017-11-11 19:35:23	RENAME_NEW_NAME(0x00002000)	74Aulovt6GcQYWWHoK6FoP XvDtaLRwZKlcSggpRHemKoQ PeCRFFqCb7VCpNoOQNI12E YF7NWE9QRxkNARDHyCivi4	Not Content Indexed (0x00002000)	24022	5	Unable to resolve path of parent entry
26928640	2017-11-11 19:35:23	RENAME_NEW_NAME(0x00002000) CLOSE(0x80000000)	74Aulovt6GcQYWWHoK6FoP XvDtaLRwZKlcSggpRHemKoQ PeCRFFqCb7VCpNoOQNI12E YF7NWE9QRxkNARDHyCivi4	Archive (0x00000020), Not Content Indexed	24022	5	Unable to resolve path of parent entry

⑧ 2017.11.11. 19:45:30 / 'Moo0 Anti Recovery 1.11' 삭제

USN	레코드 시간 정보	변경 원인 플래그	대상 파일 이름	대상 파일 타입	대상 파일의 MFT Entry 번호	대상 파일의 MFT Entry Sequence 번호	전체 경로
29583760	2017-11-11 19:45:30	FILE_DELETE(0x00000200) CLOSE(0x80000000)	AntiRecovery.exe	Archive (0x00000020)	23713	3	Unable to resolve path of parent entry
29583856	2017-11-11 19:45:30	FILE_DELETE(0x00000200) CLOSE(0x80000000)	Moo0 데이타 회복방지기 1.11.lnk	Archive (0x00000020)	23704	8	D₩Users₩Yun_Desktop₩Desktop
			~중략~				
29584544	2017-11-11 19:45:32	FILE_CREATE(0x00000100)	UNINSTALLER.EXE-326DE5BE.pf	Archive (0x00000020), Not Content Indexed	23713	4	D₩Windows₩Prefetch
29584664	2017-11-11 19:45:32	DATA_EXTEND(0x00000002) FILE_CREATE(0x00000100)	UNINSTALLER.EXE-326DE5BE.pf	Archive (0x00000020), Not Content Indexed	23713	4	D₩Windows₩Prefetch
29584784	2017-11-11 19:45:32	DATA_EXTEND(0x00000002) FILE_CREATE(0x00000100) CLOSE(0x80000000)	UNINSTALLER.EXE-326DE5BE.pf	Archive (0x00000020), Not Content Indexed	23713	4	D₩Windows₩Prefetch
29584904	2017-11-11 19:45:32	FILE_DELETE(0x00000200) CLOSE(0x80000000)	uninstaller.exe	Archive (0x00000020)	23716	3	Unable to resolve path of parent entry
29585000	2017-11-11 19:45:32	FILE_DELETE(0x00000200) CLOSE(0x80000000)	Anti-Recovery 1.11		23675	13	D₩Program Files (x86)₩Moo0

[그림6-1] 증거인멸 사례의 $UsnJrnl 레코드 정보

레코드①을 통해 2017-11-11 18:30:20 에 사용자가 '보안.patch', '증거분석결과보고서.pdf', 'K_사건관련중간보고.pptx', '보도자료_스크랩.jpg', 'K_사건관련보고사항정리.hwp'파일을 일괄삭제(File_Closed/File_Deleted)한 것을 확인할 수 있다. 5개의 파일 모두 계정명 'Yun_Desktop'의 바탕화면에 저장되어 있었다는 것도 알 수 있다.

레코드②에서는 사용자가 2017-11-11 18:40:31 에 시스템 시간을 변경한 것을 보여주고 있다. $UsnJrnl 파일의 USN은 순차적으로 저장된다는 점에서 시간의 역행 흔적을 이 파일에서도 확인해 볼 수 있을 것이다.

레코드③에서는 2017-10-11 18:41:06 에 사용자가 'K_사건관련각종보고'라는 하나의 디렉토리(File Attribute로 확인)를 삭제(File_Renamed_Old>File_Renamed_New>File_Renamed_New/File_Closed)한 것을 확인할 수 있다. 위 디렉토리가 삭제되면서 그 경로는 $Recycle.Bin\S-1-5-21-2025115636- 2423932730-2888420290-1001로 변경되는데 위 위치는 SID가 'S-1-5-21-2025115636-2423932730-2888420290-1001'인 사용자의 휴지통을 의미한다. 디렉토리가 이동(삭제)하면서 디렉토리명이 'K_사건관련각종보고' 에서 '$R09J51J'로 변경된 것도 알 수 있다.

이어지는 동일한 시간의 레코드를 살펴보면 '$R09J51J' 하위에 '월간보고.hwp', '일일업무보고.hwp', '주간보고.hwp' 파일 세 개의 이벤트가 모두 'Access_Right_Changed>Access_Right_Changed>File_Closed' 임을 확인할 수 있다.[34] 세 개의 파일 모두 위 휴지통으로 이동(삭제)되어 접근 권한이 변경된 것이다.

레코드④로부터 사용자가 2017-10-11 18:42:26 에 'eventvwr' 파일에 접근한 사실이 확인된다. EventInfo를 통해 $Data 속성에 데이터가 추가되었다는 것도 확인할 수 있는데, 이는 해당 파일에 작업이 수행됨을 의

34) 시스템은 휴지통을 공유하지 않고 사용자 계정별로 독립적으로 사용한다. 만약 다른 사용자의 휴지통 폴더로 이동을 시도하면 접근이 제한된다.

미한다고 볼 수 있겠다. 또한 같은 시간대에 접근하는 파일들의 경로를 보면 모두 이벤트 로그와 관련 있는 것으로 추정되는 경로임을 확인할 수 있다.

- ₩Users₩Yun_Desktop₩AppData₩Roaming₩Microsoft₩MMC₩eventvwr
- ₩Users₩Yun_Desktop₩AppData₩Local₩Microsoft₩Event Viewer₩RecentViews
- ₩ProgramData₩Microsoft₩Event Viewer₩Windows 로그₩Channel_0.xml
- ₩Users₩Yun_Desktop₩AppData₩Local₩Microsoft₩Event Viewer₩Settings.Xml

이어서 2017-10-11 18:43:16 에 프리패치 파일 'DFRGUI.EXE-C853 DD35.pf'가 실행된 것이 확인된다. 위 파일은 디스크 조각 모음 작업 시 실행되는 프리패치 파일이다.

레코드⑤에서 다시 시스템 시간이 변경된 것이 확인된다. 이때 시스템은 'NetworkService' 폴더에 접근하고 있는데 이를 통해 해당 작업이 네트워크를 통해 이루어짐을 유추해 볼 수 있다. 윈도우즈의 날짜 및 시간 변경 서비스를 사용하면서 '인터넷 시간으로 자동 업데이트' 한 것으로 보인다.

레코드⑥에서는 'Moo0 Anti Recovery 1.11'를 다운로드 받은 흔적이 확인된다. 다운로드 후 설치 직전에 'Moo0_ANTI_RECOVERY_V1.11_INST-1C771390.pf' 파일이 생성되고 'Program Files (x86)' 경로에 'Moo0'라는 디렉토리와 'Anti-Recovery 1.11'가 설치 되었다. 또한 바탕화면에 'Moo0 데이타 회복방지기 1.11.lnk' 바로가기 파일이 생성된 것을 볼 수 있다.

레코드⑦은 'Moo0 Anti Recovery' 프로그램이 실행된 것으로 보이는 화면이다. 실행 전 'ANTIRECOVERY.EXE-AF101B70.pf' 라는 프리패치 파일이 실행되고 숫자 0에서 139까지의 파일과 0으로 덮어쓴 파일(Archive)이 생성되는 것을 볼 수 있다. 이어서 0에서 139까지의 파일 속성 정보가 변경되고 숫자로 변경된 파일들은 다시 '000...'으로 덮어쓰이며 또 다시 임의의 문자열로 변경되는 것이 확인된다.

레코드⑧에서는 프로그램 삭제 시 실행파일과 링크파일이 함께 삭제되고, Uninstall 프리패치 파일 'UNINSTALLER.EXE-326DE5BE.pf'이 실행되면서 'Program Files(x86)' 내의 설치 프로그램이 삭제되는 것을 확인할 수 있다.

(2) 단서로서의 $UsnJrnl 파일 활용

앞서 분석한 $UsnJrnl 파일의 각 레코드에서 발견된 단서를 시작으로 해당 경로 등에 접근해 보면서 실제 $UsnJrnl 파일을 활용할 수 있는 방법을 제안하고자 한다. 이때 각각의 시간 정보를 비교하여 $UsnJrnl 파일의 시간 정보의 신뢰성도 확인해 볼 것이다.

① 삭제 파일 흔적 추적

기본적으로 포렌식 도구들은 [그림6-3]과 같이 삭제된 파일들을 복구해 그 목록을 보여주는데, 위 케이스의 경우 [그림6-2]과 같이 파일명이 모두 임의의 문자로 변경되어 파일명을 통해서는 중요(혐의 관련) 문서를 특정하기 어렵다. 이는 삭제 프로그램의 흔적으로 판단되며, 삭제 프로그램마다 파일을 삭제하는 방식에 차이가 있는데 해당 시나리오에서 사용한 파일 삭제 프로그램 'Moo0'는 '파일이름삭제' 옵션을 선택하는 경우 삭제 이전 파일명을 확인할 수 없도록 파일명을 변경하는 것을 알 수 있다.

	Name	Logical Size	File Ext	Description	Category	Entry Modified
1	fPG7KMwvirKSkYwtO3k4WLOjaI7qYKcX...	0		File, Deleted	Unknown	2017-11-11 19:35:23
2	RelImDr2me9Jih9zwFbOR4TybCMBQMS...	0		File, Deleted	Unknown	2017-11-11 19:35:23
3	Y2XD7JwC9zWJSrtznk8pLHB1vJpOHfdFI...	0		File, Deleted	Unknown	2017-11-11 19:35:23
4	9I3zT4kyQ6CiVBmPuMMXK1xa2IGPxPnQ...	0		File, Deleted	Unknown	2017-11-11 19:35:23
5	a9hbEApgCohVUfIa2IpipQw6pDhhZY6y...	0		File, Deleted	Unknown	2017-11-11 19:35:23
6	ErWyBuGzhVUPwcqsvGokFvfKknGfsrz6...	0		File, Deleted	Unknown	2017-11-11 19:35:23
7	B8pDXqKqWoWRMFC2WVPLWJ9Isruap...	0		File, Deleted	Unknown	2017-11-11 19:35:23
8	EXuM9v7bjaUyG2k1KC6xF5dvkUNFctJVY...	0		File, Deleted	Unknown	2017-11-11 19:35:23
9	gLjJcFU3MOkUwDqx7uO4cXb9nZyyqQ7...	0		File, Deleted	Unknown	2017-11-11 19:35:23
10	jmHzcd8RyU9OBrRMsECxh5tcPeOYC6Ea...	0		File, Deleted	Unknown	2017-11-11 19:35:23

[그림6-2] 삭제 프로그램을 사용한 증거인멸 케이스의 Lost Files 폴더 리스트

	Name	Logical Size	File Ext	Description	Category	Entry Modified
1	#www.credu.com	48	com	Folder, Deleted, Not Index...	Folder	2017-09-01 13:23:41
2	02	48		Folder, Deleted, Not Index...	Folder	2017-08-07 22:54:23
3	_metadata	48		Folder, Deleted, Not Index...	Folder	2017-08-14 08:35:49
4	_platform_specific	48		Folder, Deleted, Not Index...	Folder	2017-08-14 08:35:50
5	gu-in	48		Folder, Deleted	Folder	2017-09-01 13:34:49
6	he-il	48		Folder, Deleted	Folder	2017-09-01 13:34:49
7	hi-in	48		Folder, Deleted	Folder	2017-09-01 13:34:49
8	hr-hr	48		Folder, Deleted	Folder	2017-09-01 13:34:49
9	hu-hu	48		Folder, Deleted	Folder	2017-09-01 13:34:49
10	id-id	48		Folder, Deleted	Folder	2017-09-01 13:34:49

[그림6-3] 일반적인 케이스의 Lost Files 폴더 리스트

파일 삭제 행위는 실제 데이터에 대한 삭제가 아닌 해당 파일의 MFT Entry 정보를 삭제(변경)함으로써 실제 데이터에 대한 위치를 포함한 정보를 지우게 된다. 만약 이 MFT Entry가 다른 파일에 재할당되어 덮어씌워지면 복구도 불가능하다. 하지만 MFT Entry가 다른 파일에 의해 덮어씌워진 경우에도 이전의 MFT Entry 정보를 가진 $UsnJrnl 레코드가 존재한다면 삭제된 파일의 정보를 추적해 볼 수 있다.

아래의 [그림6-4]은 임의의 문자열로 파일명이 변경된 삭제 파일 중 MFT Entry 번호 '64183'을 사용한 레코드들만을 보여준 그림이다. USN 번호와 대상 파일의 MFT Entry Sequence 번호를 기준으로 해당 MFT Entry가 재할당된 순서를 확인해 보면 '증거분석결과보고서.pdf'가 삭제(2017-11-11 18:30:20)된 후 'testTempViewerDir' 파일에 재할당 되었고,

해당 파일이 삭제(2017-10-11 18:41:21)된 후 다시 'RecoveryStore.{8570701D-C6C7-11E7-AF45-1078D2F7FB8A}.dat' 파일에 재할당 되었다. 이어서 해당 파일 삭제(2017-11-11 19:06:19) 후 'Moo0' 프로그램에 의해 파일명이 '30'으로 변경되었고 다시 'wMQnQ~'로 시작되는 파일명으로 바뀐 것을 확인할 수 있다.

'testTempViewerDir' 파일과 'RecoveryStore.{8570701D-C6C7-11E7-AF45-1078D2F7FB8A}.dat'파일은 시스템에 의해 자동 생성되고 삭제되는 파일이므로 실제 사용자가 직접 삭제한 파일은 '증거분석결과보고서.pdf'일 것이다.

USN	레코드 시간 정보	변경 원인 플래그	대상 파일 이름	대상 파일 타입	대상 파일의 MFT Entry 번호	대상 파일의 MFT Entry Sequence 번호
26337280	2017-11-11 18:30:20	FILE_DELETE(0x00000200) CLOSE(0x80000000)	증거분석결과보고서.pdf	Archive (0x00000020)	64183	9
26344104	2017-10-11 18:41:21	FILE_CREATE(0x00000100)	testTempViewerDir	Not Content Indexed (0x00002000)	64183	10
26344200	2017-10-11 18:41:21	FILE_CREATE(0x00000100) CLOSE(0x80000000)	testTempViewerDir	Not Content Indexed (0x00002000)	64183	10
26344296	2017-10-11 18:41:21	FILE_DELETE(0x00000200) CLOSE(0x80000000)	testTempViewerDir	Not Content Indexed (0x00002000)	64183	10
26349856	2017-11-11 19:03:10	FILE_CREATE(0x00000100)	RecoveryStore.{8570701D-C6C7-11E7-AF45-1078D2F7FB8A}.dat	Archive (0x00000020), Not Content Indexed	64183	11
26350032	2017-11-11 19:03:10	DATA_EXTEND(0x00000002) FILE_CREATE(0x00000100)	RecoveryStore.{8570701D-C6C7-11E7-AF45-1078D2F7FB8A}.dat	Archive (0x00000020), Not Content Indexed	64183	11
26350208	2017-11-11 19:03:10	DATA_OVERWRITE(0x00000001) DATA_EXTEND(0x00000002) FILE_CREATE(0x00000100)	RecoveryStore.{8570701D-C6C7-11E7-AF45-1078D2F7FB8A}.dat	Archive (0x00000020), Not Content Indexed	64183	11
26350720	2017-11-11 19:03:10	DATA_OVERWRITE(0x00000001) DATA_EXTEND(0x00000002) DATA_TRUNCATION(0x00000004) FILE_CREATE(0x00000100)	RecoveryStore.{8570701D-C6C7-11E7-AF45-1078D2F7FB8A}.dat	Archive (0x00000020), Not Content Indexed	64183	11
26573192	2017-11-11 19:06:19	DATA_OVERWRITE(0x00000001) DATA_EXTEND(0x00000002) DATA_TRUNCATION(0x00000004) FILE_CREATE(0x00000100)	RecoveryStore.{8570701D-C6C7-11E7-AF45-1078D2F7FB8A}.dat	Archive (0x00000020), Not Content Indexed	64183	11
26573368	2017-11-11 19:06:19	FILE_DELETE(0x00000200) CLOSE(0x80000000)	RecoveryStore.{8570701D-C6C7-11E7-AF45-1078D2F7FB8A}.dat	Archive (0x00000020), Not Content Indexed	64183	11

26643456	2017-11-11 19:35:22	FILE_CREATE(0x00000100)	30	Archive (0x00000020)	64183	12
26643968	2017-11-11 19:35:22	FILE_CREATE(0x00000100) CLOSE(0x80000000)	30	Archive (0x00000020)	64183	12
27319808	2017-11-11 19:35:24	BASIC_INFO_CHANGE(0x00008000)	30	Normal (0x00000080)	64183	12
27320320	2017-11-11 19:35:24	BASIC_INFO_CHANGE(0x00008000) CLOSE(0x80000000)	30	Normal (0x00000080)	64183	12
27320832	2017-11-11 19:35:24	A user changed the FILE_ATTRIBUTE_NOT_CONTENT_INDEXED attributeINDEXABLE_CHANGE(0x00004000) BASIC_INFO_CHANGE(0x00008000)	30	Not Content Indexed (0x00002000)	64183	12
27321344	2017-11-11 19:35:24	A user changed the FILE_ATTRIBUTE_NOT_CONTENT_INDEXED attributeINDEXABLE_CHANGE(0x00004000) BASIC_INFO_CHANGE(0x00008000)	30	Not Content Indexed (0x00002000)	64183	12
27321856	2017-11-11 19:35:24	A user changed the FILE_ATTRIBUTE_NOT_CONTENT_INDEXED attributeINDEXABLE_CHANGE(0x00004000) BASIC_INFO_CHANGE(0x00008000)	30	Normal (0x00000080)	64183	12
27322368	2017-11-11 19:35:24	A user changed the FILE_ATTRIBUTE_NOT_CONTENT_INDEXED attributeINDEXABLE_CHANGE(0x00004000) BASIC_INFO_CHANGE(0x00008000)	30	Normal (0x00000080)	64183	12
27322880	2017-11-11 19:35:24	A user changed the FILE_ATTRIBUTE_NOT_CONTENT_INDEXED attributeINDEXABLE_CHANGE(0x00004000) BASIC_INFO_CHANGE(0x00008000)	30	Not Content Indexed (0x00002000)	64183	12
27323392	2017-11-11 19:35:24	A user changed the FILE_ATTRIBUTE_NOT_CONTENT_INDEXED attributeINDEXABLE_CHANGE(0x00004000) BASIC_INFO_CHANGE(0x00008000)	30	Not Content Indexed (0x00002000)	64183	12

27323904	2017-11-11 19:35:24	RENAME_OLD_NAME(0x00001000)	30	Not Content Indexed (0x00002000)	64183	12
27324416	2017-11-11 19:35:24	RENAME_NEW_NAME(0x00002000)	wMQnQlWiTOL73kMjnvSl4gRdlTy9 9rlj5ka3sH7BFVYoNmE9cfg6jsBPq 22bUHwEWgPDra5As53ljQAIK5LTK xBlPAkImBtDRA2k59qsp254ESVp3	Not Content Indexed (0x00002000)	64183	12
27324928	2017-11-11 19:35:24	RENAME_NEW_NAME(0x00002000) CLOSE(0x80000000)	wMQnQlWiTOL73kMjnvSl4gRdlTy9 9rlj5ka3sH7BFVYoNmE9cfg6jsBPq 22bUHwEWgPDra5As53ljQAIK5LTK xBlPAkImBtDRA2k59qsp254ESVp3	Archive (0x00000020), Not Content Indexed	64183	12
27325440	2017-11-11 19:35:24	A user changed the FILE_ATTRIBUTE_NOT_CONTENT_INDEXED attributeINDEXABLE_CHANGE(0x00004000) BASIC_INFO_CHANGE(0x00008000)	wMQnQlWiTOL73kMjnvSl4gRdlTy9 9rlj5ka3sH7BFVYoNmE9cfg6jsBPq 22bUHwEWgPDra5As53ljQAIK5LTK xBlPAkImBtDRA2k59qsp254ESVp3	Normal (0x00000080)	64183	12
27325952	2017-11-11 19:35:24	A user changed the FILE_ATTRIBUTE_NOT_CONTENT_INDEXED attributeINDEXABLE_CHANGE(0x00004000) BASIC_INFO_CHANGE(0x00008000)	wMQnQlWiTOL73kMjnvSl4gRdlTy9 9rlj5ka3sH7BFVYoNmE9cfg6jsBPq 22bUHwEWgPDra5As53ljQAIK5LTK xBlPAkImBtDRA2k59qsp254ESVp3	Normal (0x00000080)	64183	12
27326464	2017-11-11 19:35:24	BASIC_INFO_CHANGE(0x00008000)	wMQnQlWiTOL73kMjnvSl4gRdlTy9 9rlj5ka3sH7BFVYoNmE9cfg6jsBPq 22bUHwEWgPDra5As53ljQAIK5LTK xBlPAkImBtDRA2k59qsp254ESVp3	Normal (0x00000080)	64183	12
27326976	2017-11-11 19:35:24	BASIC_INFO_CHANGE(0x00008000) CLOSE(0x80000000)	wMQnQlWiTOL73kMjnvSl4gRdlTy9 9rlj5ka3sH7BFVYoNmE9cfg6jsBPq 22bUHwEWgPDra5As53ljQAIK5LTK xBlPAkImBtDRA2k59qsp254ESVp3	Normal (0x00000080)	64183	12
27327488	2017-11-11 19:35:24	FILE_DELETE(0x00000200) CLOSE(0x80000000)	wMQnQlWiTOL73kMjnvSl4gRdlTy9 9rlj5ka3sH7BFVYoNmE9cfg6jsBPq 22bUHwEWgPDra5As53ljQAIK5LTK xBlPAkImBtDRA2k59qsp254ESVp3	Normal (0x00000080)	64183	12

[그림6-4] MFT Entry 번호 64183을 사용한 $UsnJrnl파일 레코드

사용자가 선택한 파일 삭제 방법은 '파일 이름 변경' 이었다. 다른 정보의 변경 여부를 확인하기 위해 $LogFile를 대상으로 별도 분석을 시도하였다. 이 파일에서도 역시 동일한 MFT Entry 번호를 사용하는 레코드들만을 확인하기 위해 Target VCN 번호와 MFT Cluster Index 데이터가 동일한 레코드들을 선별하였다.

그 결과, 파일 이름을 변경하는 것 외에도 시간 정보도 변경되는 것을 확인할 수 있었다.

LSN	레코드 시간 정보	이벤트	상세정보	대상 파일 이름	생성 시간	수정 시간	MFT Entry 수정시간	접근 시간	대상파일의 VCN	클러스터 인덱스
823669369	2017-11-11 18:06:06	Renaming File	Newtech.pdf -> 증거분석결과보고서.pdf	증거분석결과보고서.pdf					0x3ead	6
823926078		File Deletion		증거분석결과보고서.pdf	2017-11-10 23:54:56	2017-10-29 17:15:16	2017-11-11 18:06:06	2017-11-10 23:54:56	0x3ead	6
823968348	2017-10-11 18:41:21	Directory Creation		testTempViewerDir	2017-10-11 18:41:21	2017-10-11 18:41:21	2017-10-11 18:41:21	2017-10-11 18:41:21	0x3ead	6
823968714		File Deletion		testTempViewerDir	2017-10-11 18:41:21	2017-10-11 18:41:21	2017-10-11 18:41:21	2017-10-11 18:41:21	0x3ead	6
824087968				th[5].jpg	2017-11-09 15:03:09	2017-11-09 15:03:09	1601-01-01 22:21:43	2017-11-09 15:03:09	0x3ead	4
824212560	2017-11-11 19:03:10	File Creation		RecoveryStore.{8570701D-C6C7-11E7-AF45-	2017-11-11 19:03:10	2017-11-11 19:03:10	2017-11-11 19:03:10	2017-11-11 19:03:10	0x3ead	6
824212843		Writing Content of Non-Resident File	Cluster Number : 1840546(1)	RecoveryStore.{8570701D-C6C7-11E7-AF45-					0x3ead	6
824822510	2017-11-11 19:35:22	File Creation		30	2017-11-11 19:35:22	2017-11-11 19:35:22	2017-11-11 19:35:22	2017-11-11 19:35:22	0x3ead	6
825255409		Renaming File	000000000000000000000000000000000000000000	wMQnQlWiTOL73kMjnvSl4gRdlTy99rlj5ka3sH7REVYo					0x3ead	6
825257007		File Deletion		wMQnQlWiTOL73kMjnvSl4gRdlTy99rlj5ka3sH7REVYo	2016-10-03 01:23:37	2016-10-22 13:45:28	2017-11-11 19:35:24	2016-10-31 05:09:37	0x3ead	6

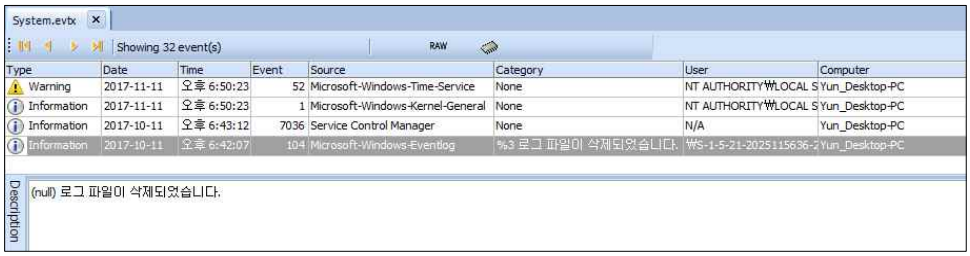
[그림6-5] MFT Entry 번호 64183을 사용한 $LogFile 레코드

② 이벤트 로그 분석

$UsnJrnl 레코드를 통해 두 번의 시스템 변경과, 디스크 조각 모음 프리 패치 실행, 이벤트 뷰어 관련 파일이나 디렉토리에 접근한 흔적을 확인할 수 있었다. 이에 이벤트 로그 분석을 통해 해당 사실들을 확인해 보도록 하겠다.

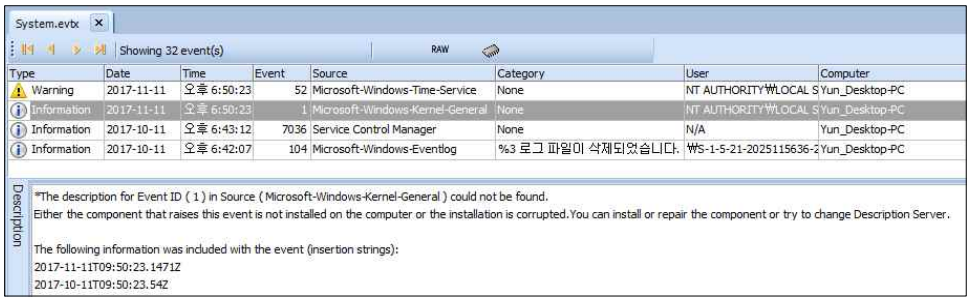
해당 시나리오 이미지로부터 'system.evtx' 파일을 추출하여 윈도우에서 제공하는 이벤트 뷰어 어플리케이션을 통해 이벤트 로그 내역을 확인한 결과, [그림6-6]과 [그림6-7],[그림6-8]에서처럼 '이벤트 로그 삭제' 사실과 '시스템 시간 변경' 사실을 확인할 수 있다. 여기서 발견된 것은 '2017.10.11. 18:43:26'에서 '2017.11.11. 18:50:59'로 변경된 정보만 확인이 된다. 이벤트 로그 삭제와 함께 이전에 변경한 시스템 시간 변경 로그가 삭제된 것이다. 실제 사례에서는 삭제된 이벤트 로그에 대한 파일 카빙 등의 작업을 통해 이전의 이벤트 로그들을 분석해 볼 필요가 있겠다.

두 번째 시스템 시간 변경의 경우 7초 가량의 시간이 소요되었다. 이것은 시간을 업데이트 하면서 소요된 시간으로 보인다. 또한 'NetworkService' 폴더에 접근하는 것을 통해 네트워크를 이용한 시간 변경을 한 것을 추측해 볼 수 있다.

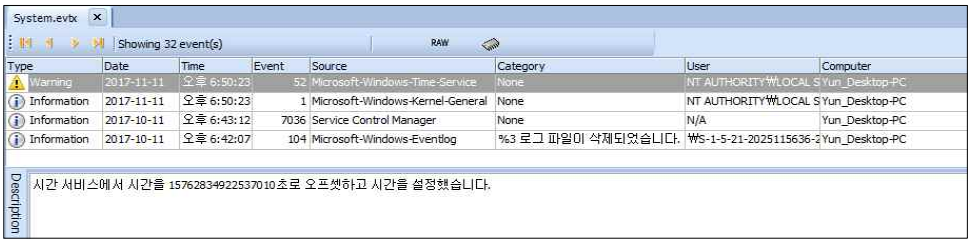


[그림6-6] 이벤트 로그 삭제 흔적

35) 목록상의 Date 시간과 하단의 Description 시간이 차이가 있는데 이는 UTC+00:00 기준 시간을 표기한 것으로 한국 시간으로 변경하려면 9시간을 더해주어야 한다.

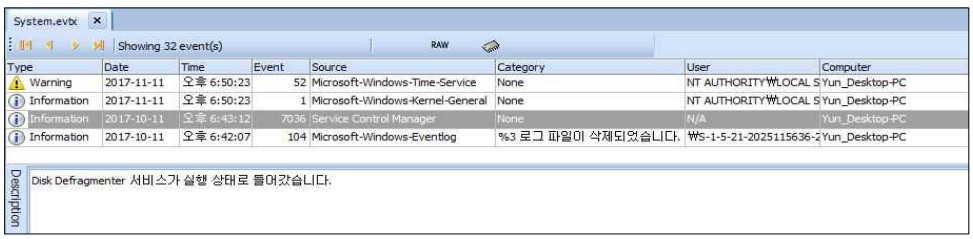


[그림6-7] 시스템 시간 변경 흔적 1[35)]



[그림6-8] 시스템 시간 변경 흔적 2

앞선 레코드 일괄 분석 시 디스크 조각 모음 실행 이벤트가 발생했음을 유추할 수 있는 정보들도 확인이 되는데 [그림6-9]에서처럼 이벤트 로그에서 실행 상태를 기록(2017-10-11 18:43:12)하고 프리패치를 실행(2017-10-11 18:43:16)하는 것이 확인된다.



[그림6-9] 디스크 조각 모음 이벤트 실행 흔적

③ 3개의 문건이 들어있는 1개의 디렉토리 삭제 흔적

레코드③에서 'K_사건관련각종보고' 디렉토리와 '월간보고.hwp', '일일

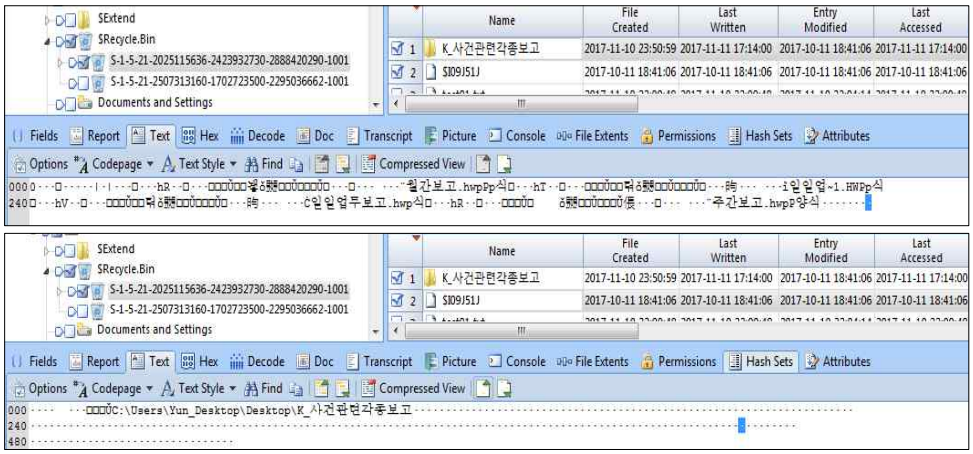
업무보고.hwp', '주간보고.hwp' 파일이 동일한 시간에 삭제된 것이 확인된다. '월간보고.hwp', '일일업무보고.hwp', '주간보고.hwp' 파일들의 경로 정보를 통해 해당 파일들은 'K_사건관련각종보고' 디렉토리 하위에 존재했던 것도 알 수 있다. 위의 디렉토리가 '$R09J51J'로 변경되면서 그 경로가 'Recycle.Bin'으로 변경되었으므로 사용자의 휴지통 디렉토리를 확인해 볼 수 있겠다.

먼저 휴지통의 SID와 위 케이스 사용자의 SID를 비교해 본 결과 그 값이 일치하였다.

	User	SID	CollectionTime	Artifact Path	Job	Target
1	Administrator	S-1-5-21-2025115636-2423932730-2888420290-500	2017-11-14 02:06:03	Registry	EvProc 20171113170603	증거인멸사례
2	Guest		2017-11-14 02:06:03	Registry	EvProc 20171113170603	증거인멸사례
3	HomeGroupUser$	S-1-5-21-2025115636-2423932730-2888420290-1002	2017-11-14 02:06:03	Registry	EvProc 20171113170603	증거인멸사례
4	S-1-5-18	S-1-5-18	2017-11-14 02:06:03	Registry	EvProc 20171113170603	증거인멸사례
5	S-1-5-19	S-1-5-19	2017-11-14 02:06:03	Registry	EvProc 20171113170603	증거인멸사례
6	S-1-5-20	S-1-5-20	2017-11-14 02:06:03	Registry	EvProc 20171113170603	증거인멸사례
7	Yun_Desktop	S-1-5-21-2025115636-2423932730-2888420290-1001	2017-11-14 02:06:03	Registry	EvProc 20171113170603	증거인멸사례

[그림6-10] 사용자의 SID 확인

하지만 해당 휴지통에서 삭제된 디렉토리는 확인되나 해당 디렉토리 하위에 존재하던 파일들은 휴지통 목록에 존재하지 않는다. 다만, 삭제된 디렉토리의 인덱스 정보와 $I09J51J(원본 파일의 경로 정보를 저장)파일을 통해 'K_사건관련각종보고' 디렉토리는 'C:\Users\Yun_Desktop\Desktop\'에 존재했었고, 그 하위에는 '월간보고.hwp', '일일업무보고.hwp', '주간보고.hwp'파일이 존재했던 것을 알 수 있는데, 휴지통에 존재하는 기록들과 $UsnJrnl 레코드를 함께 분석하면 하위 파일들이 'K_사건관련각종보고' 디렉토리가 삭제되면서 일괄삭제 되었다는 구체적 사실을 추가로 확인해 볼 수 있겠다.



[그림6-11] 휴지통 내 삭제된 'K_사건관련각종보고' 디렉토리 흔적

④ 'Moo0 Anti Recovery 1.11' 다운로드(미실행)

$UsnJrnl 레코드에서 사용자가 'Moo0 Anti Recovery 1.11'를 다운로드 받은 흔적을 발견할 수 있었다. 실제로 다운로드 받은 경로와 프로그램을 설치한 경로를 찾아가 보았다.

	Name	File Created	Last Written	Entry Modified	Last Accessed
1	Moo0_Anti_Recovery_v1.11_Installer.exe	2017-11-11 19:05:13	2017-11-11 19:05:14	2017-11-11 19:05:14	2017-11-11 19:05:14
2	Moo0_Anti_Recovery_v1.11_Installer.exe:Zone.Identifier				
3	desktop.ini	2017-11-07 23:28:36	2017-11-07 23:28:41	2017-11-07 23:28:41	2017-11-07 23:28:36

[그림6-12] Moo0 Anti Recovery 1.11 설치 파일 흔적

위 그림을 통해 다운로드 받은 시간(File Created) 시간과 $UsnJrnl에서 기록된 시간이 일치함을 확인할 수 있다. 또한 프리패치 생성 흔적도 확인할 수 있는데 그 시간 정보도 일치한다.

	Name	File Created	Last Written	Entry Modified	Last Accessed
58	MOO0_ANTI_RECOVERY_V1.11_INST-1C771390.pf	2017-11-11 19:05:36	2017-11-11 19:05:36	2017-11-11 19:05:36	2017-11-11 19:05:36

[그림6-13] Moo0 Anti Recovery 1.11 프리패치 실행 흔적

사용자가 경로를 변경하지 않으면 기본적으로 프로그램 파일들은

'Program Files (x86)'의 경로에 설치된다. 해당 사례에서도 이 경로를 확인해 보면 'Moo0'라는 디렉토리를 확인할 수 있으나 현재 그 하위에는 실행 프로그램이 존재하지 않는다. 또한 설치 시 바탕화면에 바로가기('Moo0 데이타 회복방지기 1.11.lnk')를 설정했으나 현재 바탕화면에는 이 링크파일이 존재하지 않음을 확인할 수 있다. 이는 이후 수행된 프로그램 삭제의 결과임을 알 수 있다.

⑤ 'Moo0 Anti Recovery 1.11' 프로그램 실행

위 프로그램이 실행될 때 'ANTIRECOVERY.EXE-AF101B70.pf' 프리패치 파일이 실행되는 것을 위 $UsnJrnl 레코드에서 확인할 수 있었다. 실제 해당 프리패치 파일의 생성 시간을 확인해 보면 시간 정보가 $UsnJrnl에서 확인된 실행 시간인 '2017.11.11. 19:35:16'와 일치한다.

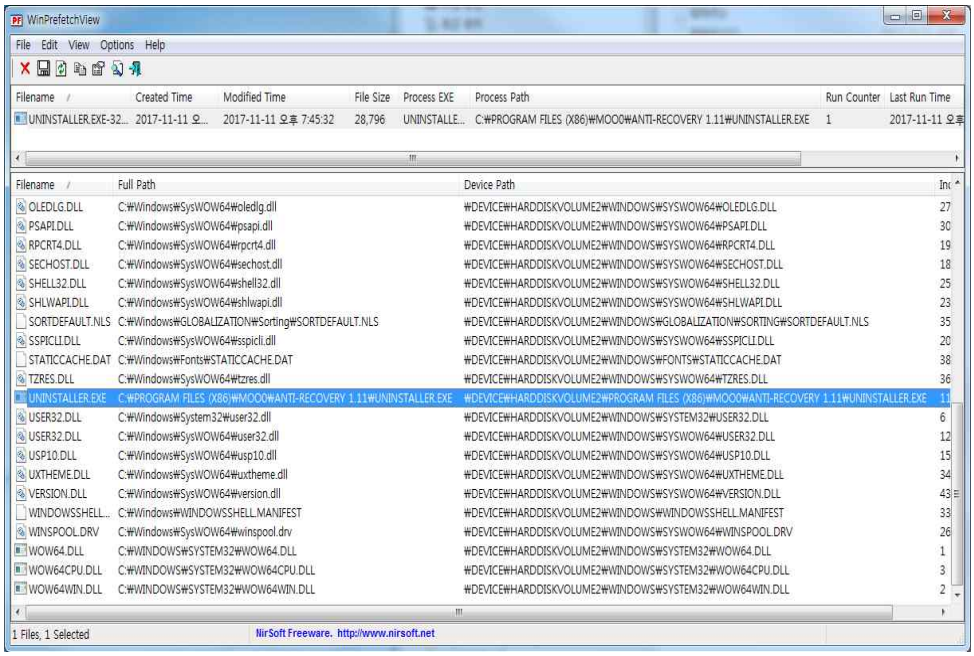
	Name	File Created	Last Written	Entry Modified	Last Accessed
10	ANTIRECOVERY.EXE-AF101B70.pf	2017-11-11 19:35:16	2017-11-11 19:35:16	2017-11-11 19:35:16	2017-11-11 19:35:16

[그림6-14] ANTIRECOVERY.EXE-AF101B70.pf 프리패치파일의 시간 정보

실행 후 삭제 파일 리스트들은 임의의 문자열로 변경되는 것을 위(1)의 레코드⑦에서 확인하였다. 실제 삭제 파일들의 목록을 보여주는 Lost Files 디렉토리를 확인해 보면 임의의 문자열로 된 파일 목록을 확인할 수 있다.

⑥ 'Moo0 Anti Recovery 1.11' 삭제(프로그램 실행 및 삭제)

프로그램을 삭제하는 경우에는 'UNINSTALLER.EXE-326DE5BE.pf'이 실행되는 것을 볼 수 있는데, 해당 프리패치 파일 뷰어를 통해 실제 Device Path로 이를 확인할 수 있다.



[그림6-15] UNINSTALLER.EXE-326DE5BE.pf 프리패치 실행 흔적

2. 기술유출 사례 분석

두 번째는 기술유출 사례이다. 기술유출 사례의 경우 해당 파일이 유출된 경로나 그 시간이 중점 분석 대상이 된다. 기술 유출 사건은 피해자의 기술이 목적 외로 사용됨에 앞서 해당 기술을 소지하고 있는 것만으로도 혐의가 인정되므로 현재 피의자 소유의 기기에 대한 분석도 반드시 필요하나, 아래의 사례는 피의자가 피해자의 회사에서 발생시킨 이벤트와 그 유출 경로를 확인하고자 함으로 그 대상을 피해자의 회사에서 사용하던 기기만으로 한정하였다.

시 나 리 오
피고인은 현재 K회사의 연구개발팀 수석 연구원으로 재직중이다. 피고인은 이직 당시 이전에 근무하던 L회사의 핵심 기술 정보를 제공 받는 조건으로 K회사의 수석 연구원으로 입사하기로 하여, 퇴사 전 L회사의 핵심 기술인 'BMX-S500J'에 대한 기술 자료와 도면, 기타 장부 내역을 관리하던 엑셀 파일을 빼내오기로 마음먹고, 다음과 같이 기술 유출을 시도하였다. - 'BMX-S500J.xlsx'내 회사 이름을 K회사로 변경 - 'BMX-S500J.xlsx'에서 'Project_A505.xlsx'로 파일 이름 변경 - 'NewFileTime' 프로그램을 이용하여 'Project_A505.xlsx' 생성일자를 '2015-11-10 16:50:26'로 변경 - 자신의 iCloud 계정을 통해 'Project_A505.xlsx' 파일 업로드 - 완전 삭제 프로그램인 'Eraser' 프로그램을 이용하여 'Project_A505.xlsx' 삭제 ※ 수사기관에서 확보된 정보 : 유출된 기술 프로젝트명 'BMX-S500J'

1) 분석 환경 및 분석 방법

(1) 가상 시나리오 시스템 정보 및 사용 도구

대상	내용
파일명	가상시나리오_기술유출.E01
운영체제	Windows 8.1 Pro K
파일시스템	NTFS
표준시간대	UTC +09:00
증거사본작성 및 파일 추출, 분석 도구	EnCase v7.12.01
$UsnJrnl 분석 도구	NTFS Log Tracker, NTFS $UsnJrnl Parser(v5.0.1)

(2) 가상 시나리오 타임라인 (시스템 시간 기준)

시간	행위 → 방법
2017.11.27. 15:47 경	파일 내용 변경 → BMX-S500J.xlsx의 두 개의 시트에서 하나의 셀씩 변경(문자의 크기 동일)
2017.11.27. 16:08 경	파일 이름 변경 → BMX-S500J.xlsx > Project_A505.xlsx
2017.11.27. 16:33 경	시간 정보 변경 → New File Time[36] 프로그램으로 수정, 생성, 접근 시간 변경
2017.11.27. 16:40 경	iCloud[37]로 파일 'Project_A505.xlsx' 업로드 → 즐겨찾기 되어 있던 iCloud에 로그인 후 파일 업로드
2017.11.27. 17:01 경	'Project_A505.xlsx' 삭제 → 파일 삭제 프로그램 'Eraser[38]'로 'Gutmann[39](35passes)' 선택 후 삭제

36) Windows 시스템 내의 파일들에 대한 시간 정보 수정 프로그램, Nenad Hrg 개발, 사용버전 NewFileTime v3.0.1.0

37) 애플에서 제공하는 클라우드 컴퓨팅 서비스로 마이크로소프트 윈도 운영체제의 컴퓨터 등의 다수의 장비에 다운로드 하며 공유할 수 있다. 위키백과, '아이클라우드' https://ko.wikipedia.org/wiki/%EC%95%84%EC%9D%B4%ED%81%B4%EB%9D%BC%EC%9A%B0%EB%93%9C, 2017.11.30.

38) 하드디스크에 저장된 데이터를 완전히 지워 소프트웨어적, 하드웨어적 방법으로 복구가 불가능하게 하는 프로그램, The Eraser Project 개발, 사용버전 Eraser 6.2.0.2979

2) 분석 결과 및 활용 방법

(1) $UsnJrnl 파일을 통한 사용자 행위 추적

시간 / 행위
레코드

① 2017.11.27.15:47:07 / 파일 내용 변경

LSN	레코드 시간 정보	변경 원인 플래그	대상 파일 이름	대상 파일 타입	대상 파일의 MFT Entry 번호	대상 파일의 MFT Entry Sequence 번호	전체 경로
13221888	2017-11-27 15:47:07	SECURITY_CHANGE(0x00000800) BASIC_INFO_CHANGE(0x00008000)	DA1B3100	Archive (0x00000020)	84761	3	Users₩YUN₩Desktop₩BMX_₩개인
13221968	2017-11-27 15:47:07	OBJECT_ID_CHANGE(0x00080000)	BMX-S500J.xlsx	Archive (0x00000020)	83939	1	Users₩YUN₩Desktop₩BMX_₩개인
13222056	2017-11-27 15:47:07	SECURITY_CHANGE(0x00000800) BASIC_INFO_CHANGE(0x00008000) OBJECT_ID_CHANGE(0x00080000)	DA1B3100	Archive (0x00000020)	84761	3	Users₩YUN₩Desktop₩BMX_₩개인
13222136	2017-11-27 15:47:07	RENAME_OLD_NAME(0x00001000) OBJECT_ID_CHANGE(0x00080000)	BMX-S500J.xlsx	Archive (0x00000020)	83939	1	Users₩YUN₩Desktop₩BMX_₩개인

39) The Gutmann method is an algorithm for securely erasing the contents of computer hard disk drives, such as files. Devised by Peter Gutmann and Colin Plumb and presented in the paper Secure Deletion of Data from Magnetic and Solid-State Memory in July 1996, it involved writing a series of 35 patterns over the region to be erased., 위키피디아 https://en.wikipedia.org/wiki/Gutmann_method, 2017.11.30.

② 2017.11.27. 16:08:40 / 파일 이름 변경

LSN	레코드 시간 정보	변경 원인 플래그	대상 파일 이름	대상 파일 타입	대상 파일의 MFT Entry 번호	대상 파일의 MFT Entry Sequence 번호	전체 경로
13234528	2017-11-27 16:08:40	RENAME_OLD_NAME(0x00001000)	BMX-S500J.xlsx	Archive (0x00000020)	83939	2	Users₩YUN₩Desktop₩BMX_₩개인
13234616	2017-11-27 16:08:40	RENAME_NEW_NAME(0x00002000)	Project_A505.xlsx	Archive (0x00000020)	83939	2	Users₩YUN₩Desktop₩BMX_₩개인
13234712	2017-11-27 16:08:40	RENAME_NEW_NAME(0x00002000) CLOSE(0x80000000)	Project_A505.xlsx	Archive (0x00000020)	83939	2	Users₩YUN₩Desktop₩BMX_₩개인

③ 2017.11.27. 16:33:35~16:35:10 / 시간 정보 변경

LSN	레코드 시간 정보	변경 원인 플래그	대상 파일 이름	대상 파일 타입	대상 파일의 MFT Entry 번호	대상 파일의 MFT Entry Sequence 번호	전체 경로
13242184	2017-11-27 16:33:35	DATA_OVERWRITE(0x00000001)	NewFileTime.ini	Archive (0x00000020)	83865	2	Users₩YUN₩AppData₩Roaming₩NewFileTime
13242368	2017-11-27 16:33:35	DATA_OVERWRITE(0x00000001) CLOSE(0x80000000)	NewFileTime.ini	Archive (0x00000020)	83865	2	Users₩YUN₩AppData₩Roaming₩NewFileTime
13242464	2017-11-27 16:33:35	DATA_OVERWRITE(0x00000001)	NewFileTime.ini	Archive (0x00000020)	83865	2	Users₩YUN₩AppData₩Roaming₩NewFileTime
13242560	2017-11-27 16:33:35	DATA_OVERWRITE(0x00000001) CLOSE(0x80000000)	NewFileTime.ini	Archive (0x00000020)	83865	2	Users₩YUN₩AppData₩Roaming₩NewFileTime

~중략~

13246112	2017-11-27 16:33:45	DATA_TRUNCATION(0x00000004)	NEWFILETIME_X64.EXE-A814CD50.pf	Archive (0x00000020), Not Content Indexed	83868	2	Windows₩Prefetch
13246240	2017-11-27 16:33:45	DATA_EXTEND(0x00000002) DATA_TRUNCATION(0x00000004)	NEWFILETIME_X64.EXE-A814CD50.pf	Archive (0x00000020), Not Content Indexed	83868	2	Windows₩Prefetch
13246464	2017-11-27 16:33:45	DATA_EXTEND(0x00000002) DATA_TRUNCATION(0x00000004) CLOSE(0x80000000)	NEWFILETIME_X64.EXE-A814CD50.pf	Archive (0x00000020), Not Content Indexed	83868	2	Windows₩Prefetch

~중략~

13247392	2017-11-27 16:35:10	BASIC_INFO_CHANGE(0x00008000)	Project_A505.xlsx	Normal (0x00000080)	83939	2	Users₩YUN₩Desktop₩BMX_₩개인
13247488	2017-11-27 16:35:10	BASIC_INFO_CHANGE(0x00008000) CLOSE(0x80000000)	Project_A505.xlsx	Normal (0x00000080)	83939	2	Users₩YUN₩Desktop₩BMX_₩개인
13247584	2017-11-27 16:35:10	BASIC_INFO_CHANGE(0x00008000)	Project_A505.xlsx	Normal (0x00000080)	83939	2	Users₩YUN₩Desktop₩BMX_₩개인
13247680	2017-11-27 16:35:10	BASIC_INFO_CHANGE(0x00008000) CLOSE(0x80000000)	Project_A505.xlsx	Normal (0x00000080)	83939	2	Users₩YUN₩Desktop₩BMX_₩개인
13247776	2017-11-27 16:35:10	BASIC_INFO_CHANGE(0x00008000)	Project_A505.xlsx	Archive (0x00000020)	83939	2	Users₩YUN₩Desktop₩BMX_₩개인
13247872	2017-11-27 16:35:10	BASIC_INFO_CHANGE(0x00008000) CLOSE(0x80000000)	Project_A505.xlsx	Archive (0x00000020)	83939	2	Users₩YUN₩Desktop₩BMX_₩개인

[그림6-16] 기술유출 사례의 $UsnJrnl 레코드 정보

④ 2017.11.27. 16:40:54~16:41:21 / iCloud로 파일 업로드

LSN	레코드 시간 정보	변경 원인 플래그	대상 파일 이름	대상 파일 타입	대상 파일의 MFT Entry 번호	대상 파일의 MFT Entry Sequence 번호	전체 경로
13302032	2017-11-27 16:40:54	FILE_CREATE(0x00000100)	cloudkit[1].js	Archive (0x00000020), Not Content Indexed	85153	2	Users₩YUN₩AppData₩Local₩Packages₩windows_ie_ac_001₩AC₩INetCache₩FV61LO5R
13302120	2017-11-27 16:40:54	DATA_EXTEND(0x00000002) FILE_CREATE(0x00000100)	cloudkit[1].js	Archive (0x00000020), Not Content Indexed	85153	2	Users₩YUN₩AppData₩Local₩Packages₩windows_ie_ac_001₩AC₩INetCache₩FV61LO5R
13302208	2017-11-27 16:40:54	FILE_DELETE(0x00000200) CLOSE(0x80000000)	cloudkit[1].js	Archive (0x00000020), Not Content Indexed	85250	1	Users₩YUN₩AppData₩Local₩Packages₩windows_ie_ac_001₩AC₩INetCache₩QBCICMHB
~중략~							
13305296	2017-11-27 16:41:00	DATA_TRUNCATION(0x00000004)	IEXPLORE.EXE-908C99F8.pf	Archive (0x00000020), Not Content Indexed	83802	3	Windows₩Prefetch
13305408	2017-11-27 16:41:00	DATA_EXTEND(0x00000002) DATA_TRUNCATION(0x00000004)	IEXPLORE.EXE-908C99F8.pf	Archive (0x00000020), Not Content Indexed	83802	3	Windows₩Prefetch
13305520	2017-11-27 16:41:00	DATA_EXTEND(0x00000002) DATA_TRUNCATION(0x00000004) CLOSE(0x80000000)	IEXPLORE.EXE-908C99F8.pf	Archive (0x00000020), Not Content Indexed	83802	3	Windows₩Prefetch
~중략~							
13315856	2017-11-27 16:41:21	FILE_CREATE(0x00000100)	singleFileUpload[1]	Archive (0x00000020), Not Content Indexed	85261	2	Users₩YUN₩AppData₩Local₩Packages₩windows_ie_ac_001₩AC₩INetCache₩XTHI2U92
13315960	2017-11-27 16:41:21	FILE_CREATE(0x00000100) CLOSE(0x80000000)	singleFileUpload[1]	Archive (0x00000020), Not Content Indexed	85261	2	Users₩YUN₩AppData₩Local₩Packages₩windows_ie_ac_001₩AC₩INetCache₩XTHI2U92

⑤ 2017.11.27. 17:00:42~17:01:24 / 'Eraser'로 파일 삭제

LSN	레코드 시간 정보	변경 원인 플래그	대상 파일 이름	대상 파일 타입	대상 파일의 MFT Entry 번호	대상 파일의 MFT Entry Sequence 번호	전체 경로
13365688	2017-11-27 17:00:42	DATA_TRUNCATION(0x00000004)	ERASER.EXE-CE61944A.pf	Archive (0x00000020), Not Content Indexed	85158	4	Windows₩Prefetch
13365792	2017-11-27 17:00:42	DATA_EXTEND(0x00000002) DATA_TRUNCATION(0x00000004)	ERASER.EXE-CE61944A.pf	Archive (0x00000020), Not Content Indexed	85158	4	Windows₩Prefetch
13365896	2017-11-27 17:00:42	DATA_EXTEND(0x00000002) DATA_TRUNCATION(0x00000004) CLOSE(0x80000000)	ERASER.EXE-CE61944A.pf	Archive (0x00000020), Not Content Indexed	85158	4	Windows₩Prefetch

~중략~

13366768	2017-11-27 17:00:58	FILE_DELETE(0x00000200) CLOSE(0x80000000)	Project_A505.lnk	Archive (0x00000020)	85089	10	Users₩YUN₩AppData₩Roaming₩Microsoft₩Windows₩Recent
13366864	2017-11-27 17:00:58	FILE_CREATE(0x00000100)	Project_A505.lnk	Archive (0x00000020)	85089	11	Users₩YUN₩AppData₩Roaming₩Microsoft₩Windows₩Recent
13366960	2017-11-27 17:00:58	DATA_EXTEND(0x00000002) FILE_CREATE(0x00000100)	Project_A505.lnk	Archive (0x00000020)	85089	11	Users₩YUN₩AppData₩Roaming₩Microsoft₩Windows₩Recent
13367056	2017-11-27 17:00:58	DATA_EXTEND(0x00000002) FILE_CREATE(0x00000100) CLOSE(0x80000000)	Project_A505.lnk	Archive (0x00000020)	85089	11	Users₩YUN₩AppData₩Roaming₩Microsoft₩Windows₩Recent

~중략~

13367536	2017-11-27 17:01:24	DATA_OVERWRITE(0x00000001) DATA_EXTEND(0x00000002)	Project_A505.xlsx	Archive (0x00000020)	83939	2	Users₩YUN₩Desktop₩BMX_₩개인
13367632	2017-11-27 17:01:24	DATA_OVERWRITE(0x00000001) DATA_EXTEND(0x00000002) DATA_TRUNCATION(0x00000004)	Project_A505.xlsx	Archive (0x00000020)	83939	2	Users₩YUN₩Desktop₩BMX_₩개인
13367728	2017-11-27 17:01:24	DATA_OVERWRITE(0x00000001) DATA_EXTEND(0x00000002) DATA_TRUNCATION(0x00000004) CLOSE(0x80000000)	Project_A505.xlsx	Archive (0x00000020)	83939	2	Users₩YUN₩Desktop₩BMX_₩개인
13367824	2017-11-27 17:01:24	A user changed the FILE_ATTRIBUTE_NOT_CONTENT_INDEXED attributeINDEXABLE_CHANGE(0x00004000) BASIC_INFO_CHANGE(0x00008000)	Project_A505.xlsx	Not Content Indexed (0x00002000)	83939	2	Users₩YUN₩Desktop₩BMX_₩개인
13367920	2017-11-27 17:01:24	A user changed the FILE_ATTRIBUTE_NOT_CONTENT_INDEXED attributeINDEXABLE_CHANGE(0x00004000) BASIC_INFO_CHANGE(0x00008000) CLOSE(0x80000000)	Project_A505.xlsx	Not Content Indexed (0x00002000)	83939	2	Users₩YUN₩Desktop₩BMX_₩개인
13368016	2017-11-27 17:01:24	BASIC_INFO_CHANGE(0x00008000)	Project_A505.xlsx	Not Content Indexed (0x00002000)	83939	2	Users₩YUN₩Desktop₩BMX_₩개인
13368112	2017-11-27 17:01:24	BASIC_INFO_CHANGE(0x00008000) CLOSE(0x80000000)	Project_A505.xlsx	Not Content Indexed (0x00002000)	83939	2	Users₩YUN₩Desktop₩BMX_₩개인

~중략~

13368208	2017-11-27 17:01:24	RENAME_OLD_NAME(0x00001000)	Project_A505.xlsx	Not Content Indexed (0x00002000)	83939	2	Users₩YUN₩Desktop₩BMX_₩개인
13368304	2017-11-27 17:01:24	RENAME_NEW_NAME(0x00002000)	{0q{P1WQpfeMVb2ZM	Not Content Indexed (0x00002000)	83939	2	Users₩YUN₩Desktop₩BMX_₩개인
13368400	2017-11-27 17:01:24	RENAME_NEW_NAME(0x00002000) CLOSE(0x80000000)	{0q{P1WQpfeMVb2ZM	Archive (0x00000020), Not Content Indexed	83939	2	Users₩YUN₩Desktop₩BMX_₩개인

~중략~

13368688	2017-11-27 17:01:24	RENAME_OLD_NAME(0x00001000)	{0q{P1WQpfeMVb2ZM	Archive (0x00000020), Not Content Indexed	83939	2	Users₩YUN₩Desktop₩BMX_₩개인
13368784	2017-11-27 17:01:24	RENAME_NEW_NAME(0x00002000)	DIWr_`6MyUZsnvRUp	Archive (0x00000020), Not Content Indexed	83939	2	Users₩YUN₩Desktop₩BMX_₩개인
13368880	2017-11-27 17:01:24	RENAME_NEW_NAME(0x00002000) CLOSE(0x80000000)	DIWr_`6MyUZsnvRUp	Archive (0x00000020), Not Content Indexed	83939	2	Users₩YUN₩Desktop₩BMX_₩개인

~중략~

13369168	2017-11-27 17:01:24	RENAME_OLD_NAME(0x00001000)	DIWr_`6MyUZsnvRUp	Archive (0x00000020), Not Content Indexed	83939	2	Users₩YUN₩Desktop₩BMX_₩개인
13369344	2017-11-27 17:01:24	RENAME_NEW_NAME(0x00002000)	!EtiAdE d=z20xug1	Archive (0x00000020), Not Content Indexed	83939	2	Users₩YUN₩Desktop₩BMX_₩개인
13369440	2017-11-27 17:01:24	RENAME_NEW_NAME(0x00002000) CLOSE(0x80000000)	!EtiAdE d=z20xug1	Archive (0x00000020), Not Content Indexed	83939	2	Users₩YUN₩Desktop₩BMX_₩개인

~중략~

13369728	2017-11-27 17:01:24	RENAME_OLD_NAME(0x00001000)	!EtiAdE d=z20xug1	Archive (0x00000020), Not Content Indexed	83939	2	Users₩YUN₩Desktop₩BMX_₩개인
13369824	2017-11-27 17:01:24	RENAME_NEW_NAME(0x00002000)	sN5lUkhfx,u8Kc+b4	Archive (0x00000020), Not Content Indexed	83939	2	Users₩YUN₩Desktop₩BMX_₩개인
13369920	2017-11-27 17:01:24	RENAME_NEW_NAME(0x00002000) CLOSE(0x80000000)	sN5lUkhfx,u8Kc+b4	Archive (0x00000020), Not Content Indexed	83939	2	Users₩YUN₩Desktop₩BMX_₩개인

~중략~

13370208	2017-11-27 17:01:24	RENAME_OLD_NAME(0x00001000)	sN5lUkhfx,u8Kc+b4	Archive (0x00000020), Not Content Indexed	83939	2	Users₩YUN₩Desktop₩BMX_₩개인
13370304	2017-11-27 17:01:24	RENAME_NEW_NAME(0x00002000)	78Q0tYq9dccN4cCe7	Archive (0x00000020), Not Content Indexed	83939	2	Users₩YUN₩Desktop₩BMX_₩개인
13370400	2017-11-27 17:01:24	RENAME_NEW_NAME(0x00002000) CLOSE(0x80000000)	78Q0tYq9dccN4cCe7	Archive (0x00000020), Not Content Indexed	83939	2	Users₩YUN₩Desktop₩BMX_₩개인

~중략~

13370688	2017-11-27 17:01:24	RENAME_OLD_NAME(0x00001000)	78Q0tYq9dccN4cCe7	Archive (0x00000020), Not Content Indexed	83939	2	Users₩YUN₩Desktop₩BMX_₩개인
13370784	2017-11-27 17:01:24	RENAME_NEW_NAME(0x00002000)	X'12aUVZ1)gF-wksu	Archive (0x00000020), Not Content Indexed	83939	2	Users₩YUN₩Desktop₩BMX_₩개인
13370880	2017-11-27 17:01:24	RENAME_NEW_NAME(0x00002000) CLOSE(0x80000000)	X'12aUVZ1)gF-wksu	Archive (0x00000020), Not Content Indexed	83939	2	Users₩YUN₩Desktop₩BMX_₩개인

~중략~

13371168	2017-11-27 17:01:24	RENAME_OLD_NAME(0x00001000)	X'12aUVZ1)gF-wksu	Archive (0x00000020), Not Content Indexed	83939	2	Users₩YUN₩Desktop₩BMX_₩개인
13371264	2017-11-27 17:01:24	RENAME_NEW_NAME(0x00002000)	)Cz(PiLCdfz{uIAG!	Archive (0x00000020), Not Content Indexed	83939	2	Users₩YUN₩Desktop₩BMX_₩개인
13371360	2017-11-27 17:01:24	RENAME_NEW_NAME(0x00002000) CLOSE(0x80000000)	)Cz(PiLCdfz{uIAG!	Archive (0x00000020), Not Content Indexed	83939	2	Users₩YUN₩Desktop₩BMX_₩개인

레코드①을 통해 2017-11-27 15:47:07 에 'BMX-S500J.xlsx'파일의 Object ID 값이 변경되었고, 'DA1B3100'파일이 임의적으로 생성되었으며, 'BMX-S500J.xlsx' 파일이 '697CE912.tmp'로 변경되는 것 등을 확인할 수 있다.

레코드②에서는 2017-11-27 16:08:40 에 사용자에 의해 파일명이 'BMX-S500J.xlsx'에서 'Project_A505.xlsx'로 변경된 것이 확인된다.

레코드③에서는 2017-11-27 16:33:35 이후로 'NewFileTime.ini'가 실행되고 이어서 'NEWFILETIME_X64.EXE-A814CD50.pf' 프리패치가 실행되었으며 'Project_A505.xlsx' 파일의 변경 원인 플래그가 '0x00008000 (BASIC_INFO_CHANGE[40])'임이 확인된다.

레코드④를 통해 iCloud 접속 흔적을 발견할 수 있는데, 2017-11-27 16:40:54 에 'cloudkit[1].js'가 생성되었고, 16:41:21에는 파일이 업로드('singleFileUpload[1]')된 것이 확인된다.

레코드⑤에서는 2017-11-27 17:00:42 에 'ERASER.EXE-CE61944A.pf' 프리패치 파일이 실행되었고 곧 'Project_A505.lnk' 파일과 '개인.lnk' 파일이 삭제된다. '개인.lnk' 파일은 'Project_A505.xlsx' 파일이 존재했던 디렉토리의 링크파일이다. 이어서 17:01:24에 'Project_A505.xlsx'이 삭제되고 총 7번 파일명이 변경된다. 여기에서의 파일명 변경은 삭제를 의미한다.

---

40) A user has either changed one or more file or directory attributes (for example, the read-only, hidden, system, archive, or sparse attribute), or one or more time stamps

(2) 단서로서의 $UsnJrnl 파일 활용

① 파일 내용 수정 흔적 분석

일반적으로 파일의 내용이 변경되는 경우 시스템은 임의적으로 별도의 파일을 생성하여 작업 내용을 반영한 후 모든 작업이 종료된 시점의 시간을 최종적으로 기록하면서 기존의 파일에 대한 변경 정보를 저장한다. 위의 레코드에서도 이러한 흔적들이 발견되어 다음의 실험을 통해 이와 같은 시스템 흔적들이 실제 이루어지는지를 확인해 보았다.

내용변경.xlsx : 엑셀 파일 내 하나의 셀에서 '123'을 '456'으로 변경

< $UsnJrnl >

레코드 시간 정보	USN	대상 파일 이름	전체 경로	변경 원인 플래그	File Attribute
2017-11-30 16:30:56	19838821560	F3239100	₩Users₩Carl₩Desktop₩내용변경테스트₩F3239100	File_Created	Archive
~중략~					
2017-11-30 16:30:56	19838822280	내용변경.xlsx	₩Users₩Carl₩Desktop₩내용변경테스트₩내용변경.xlsx	File_Renamed_Old	Archive
2017-11-30 16:30:56	19838822360	950C22DC.tmp	₩Users₩Carl₩Desktop₩내용변경테스트₩950C22DC.tmp	File_Renamed_New	Archive
2017-11-30 16:30:56	19838822448	F3239100	₩Users₩Carl₩Desktop₩내용변경테스트₩F3239100	Attr_Changed/ Object_ID_Changed/ File_Renamed_Old/ Access_Right_Chang	Archive
2017-11-30 16:30:56	19838822528	내용변경.xlsx	₩Users₩Carl₩Desktop₩내용변경테스트₩내용변경.xlsx	Attr_Changed/ Object_ID_Changed/ File_Renamed_New/ Access_Right_Chang	Archive
2017-11-30 16:30:56	19838822608	950C22DC.tmp	₩Users₩Carl₩Desktop₩내용변경테스트₩950C22DC.tmp	Object_ID_Changed/ File_Renamed_New	Archive
2017-11-30 16:30:56	19838822696	950C22DC.tmp	₩Users₩Carl₩Desktop₩내용변경테스트₩950C22DC.tmp	Object_ID_Changed/ File_Renamed_New/ File_Closed	Archive
2017-11-30 16:30:56	19838822784	내용변경.xlsx	₩Users₩Carl₩Desktop₩내용변경테스트₩내용변경.xlsx	Attr_Changed/ Object_ID_Changed/ File_Renamed_New/ Access_Right_Chang	Archive
2017-11-30 16:30:56	19838822864	내용변경.xlsx	₩Users₩Carl₩Desktop₩내용변경테스트₩내용변경.xlsx	Access_Right_Changed	Archive
2017-11-30 16:30:56	19838822944	내용변경.xlsx	₩Users₩Carl₩Desktop₩내용변경테스트₩내용변경.xlsx	Access_Right_Changed/ File_Closed	Archive
2017-11-30 16:30:56	19838823024	950C22DC.tmp	₩Users₩Carl₩Desktop₩내용변경테스트₩950C22DC.tmp	File_Closed/ File_Deleted	Archive
2017-11-30 16:30:58	19838823112	~$내용변경.xlsx	₩Users₩Carl₩Desktop₩내용변경테스트₩~$내용변경.xlsx	File_Created/ Attr_Changed/ File_Added/ Content_Indexed_Attr	Archive/ Hidden/ Not_Content_Indexed

< $LogFile >

LSN	레코드 시간 정보	이벤트	상세정보	대상 파일 이름	전체 경로	생성 시간	수정 시간	MFT Entry 변경 시간	접근 시간
65836170760	2017-11-30 16:30:56	File Creation		F3239100	#Users#Carl#Desktop#내용변경테스트#F3239100	2017-11-30 16:30:56	2017-11-30 16:30:56	2017-11-30 16:30:56	2017-11-30 16:30:56
65836171029		Writing Content of Non-Resident File	Cluster Number : 12418620(32)	F3239100	#Users#Carl#Desktop#내용변경테스트#F3239100				
65836172224	2017-11-30 16:30:56	Renaming File	내용변경.xlsx -> 950C22DC.tmp	950C22DC.tmp	#Users#Carl#Desktop#내용변경테스트#950C22DC.tmp				
65836172550		Renaming File	F3239100 -> 내용변경.xlsx	내용변경.xlsx	#Users#Carl#Desktop#내용변경테스트#내용변경.xlsx				
65836173734				내용변경.xlsx	#Users#Carl#Desktop#내용변경테스트#내용변경.xlsx	2017-11-30 16:23:40	2017-11-30 16:30:56	2017-11-30 16:30:56	2017-11-30 16:30:56
65836173795	2017-11-30 16:30:56	File Deletion		950C22DC.tmp	#Users#Carl#Desktop#내용변경테스트#950C22DC.tmp	2017-11-30 16:23:40	2017-11-30 16:23:40	2017-11-30 16:30:56	2017-11-30 16:23:40
65836174067		File Deletion		~$내용변경.xlsx	#Users#Carl#Desktop#내용변경테스트#~$내용변경.xlsx	2017-11-30 16:30:52	2017-11-30 16:30:52	2017-11-30 16:30:52	2017-11-30 16:30:52

[그림6-17] 파일내용 수정 시 $UsnJrnl 레코드 정보

테스트용 엑셀 파일 '내용변경.xlsx'을 대상으로 하나의 셀 내용을 수정하였다. '123' 이라는 숫자를 '456'으로 변경하고 이때 발생하는 이벤트에 대해 $UsnJrnl 파일과 $LogFile을 함께 분석하였다. 그 결과 위의 [그림6-17]처럼 시스템의 작업 과정을 확인해 볼 수 있었는데, 해당 파일이 존재하는 디렉토리 안에 임의의 파일을 생성하고(위 테스트의 경우 'F3239100') '내용변경.xlsx'파일이 임의의 Temp 파일로 변경(위 테스트의 경우 '950C22DC.tmp')된다. 작업이 끝나면 다시 '내용변경.xlsx' 로 바꾸고 최종적으로 임시로 생성된 파일들인 'F3239100'과 '950C22DC.tmp'은 삭제되는 것을 확인할 수 있다. $LogFile에서도 동일한 이벤트가 확인된다.

이 테스트 결과를 가상 시나리오 테스트 결과와 비교해 보면 해당 이벤트가 무슨 작업을 진행했는지 유추해 볼 수 있다.

다만, 가상 시나리오와 테스트 파일 둘 모두에게서 구체적인 변경 내역은 확인되지 않았다.

LSN	레코드 시간 정보	변경 원인 플래그	대상 파일 이름	대상 파일 타입	대상 파일의 MFT Entry 번호	대상 파일의 MFT Entry Sequence 번호	전체 경로
13221088	2017-11-27 15:47:06	FILE_CREATE(0x00000100)	DA1B3100	Archive (0x00000020)	84761	3	Users₩YUN₩Desktop₩BMX_₩개인
~중략~							
13222136	2017-11-27 15:47:07	RENAME_OLD_NAME(0x00001000) OBJECT_ID_CHANGE(0x00080000)	BMX-S500J.xlsx	Archive (0x00000020)	83939	1	Users₩YUN₩Desktop₩BMX_₩개인
13222224	2017-11-27 15:47:07	RENAME_NEW_NAME(0x00002000) OBJECT_ID_CHANGE(0x00080000)	697CE912.tmp	Archive (0x00000020)	83939	1	Users₩YUN₩Desktop₩BMX_₩개인
13222312	2017-11-27 15:47:07	SECURITY_CHANGE(0x00000800) RENAME_OLD_NAME(0x00001000) BASIC_INFO_CHANGE(0x00008000) OBJECT_ID_CHANGE(0x00080000)	DA1B3100	Archive (0x00000020)	84761	3	Users₩YUN₩Desktop₩BMX_₩개인
13222392	2017-11-27 15:47:07	SECURITY_CHANGE(0x00000800) RENAME_NEW_NAME(0x00002000) BASIC_INFO_CHANGE(0x00008000) OBJECT_ID_CHANGE(0x00080000)	BMX-S500J.xlsx	Archive (0x00000020)	84761	3	Users₩YUN₩Desktop₩BMX_₩개인
13222480	2017-11-27 15:47:07	RENAME_NEW_NAME(0x00002000) OBJECT_ID_CHANGE(0x00080000) CLOSE(0x80000000)	697CE912.tmp	Archive (0x00000020)	83939	1	Users₩YUN₩Desktop₩BMX_₩개인
13222568	2017-11-27 15:47:07	SECURITY_CHANGE(0x00000800) RENAME_NEW_NAME(0x00002000) BASIC_INFO_CHANGE(0x00008000) OBJECT_ID_CHANGE(0x00080000)	BMX-S500J.xlsx	Archive (0x00000020)	84761	3	Users₩YUN₩Desktop₩BMX_₩개인
13222656	2017-11-27 15:47:07	SECURITY_CHANGE(0x00000800)	BMX-S500J.xlsx	Archive (0x00000020)	84761	3	Users₩YUN₩Desktop₩BMX_₩개인
13222744	2017-11-27 15:47:07	SECURITY_CHANGE(0x00000800) CLOSE(0x80000000)	BMX-S500J.xlsx	Archive (0x00000020)	84761	3	Users₩YUN₩Desktop₩BMX_₩개인

[그림6-18] 기술유출 시나리오의 내용 변경을 기록한 $UsnJrnl 레코드

② 파일 이름 변경

기술유출 가상 시나리오에서는 수사기관에서 이미 유출된 기술의 프로젝트 명을 알고 있다. 그래서 파일 이름 변경 이벤트에 대한 분석은 이미 알고 있는 특정 단어를 기준으로 분석을 시도하였다.

이미 제공된 단어인 'BMX-S500J'를 $UsnJrnl 레코드 내에서 우선 탐지한 후 해당 레코드의 MFT Entry 번호와 동일한 레코드들을 조합해 보는 방법으로 진행하였다. 검색 결과 'BMX-S500J' 파일은 MFT Entry 번호 '83939'를 사용하고 있었다.

아래의 [그림6-19]은 동일한 번호를 사용하는 레코드들만을 선택하여 Sequence 번호 순서대로 정렬한 그림이다. 단, 위의 파일 내용 수정 부분은 제외하였다.

	LSN	레코드 시간 정보	변경 원인 플래그	대상 파일 이름	대상 파일 타입	대상 파일의 MFT Entry 번호	대상 파일의 MFT Entry Sequence 번호	전체 경로
① 16:08	13234528	2017-11-27 16:08:40	RENAME_OLD_NAME(0x00001000)	BMX-S500J.xlsx	Archive (0x00000020)	83939	2	Users₩YUN₩Desktop₩BMX_₩개인
	13234616	2017-11-27 16:08:40	RENAME_NEW_NAME(0x00002000)	Project_A505.xlsx	Archive (0x00000020)	83939	2	Users₩YUN₩Desktop₩BMX_₩개인
	13234712	2017-11-27 16:08:40	RENAME_NEW_NAME(0x00002000) CLOSE(0x80000000)	Project_A505.xlsx	Archive (0x00000020)	83939	2	Users₩YUN₩Desktop₩BMX_₩개인

②
16:35

LSN	레코드 시간 정보	변경 원인 플래그	대상 파일 이름	대상 파일 타입	대상 파일의 MFT Entry 번호	대상 파일의 MFT Entry Sequence 번호	전체 경로
13247392	2017-11-27 16:35:10	BASIC_INFO_CHANGE(0x00008000)	Project_A505.xlsx	Normal (0x00000080)	83939	2	Users₩YUN₩Desktop₩BMX_₩개인
13247488	2017-11-27 16:35:10	BASIC_INFO_CHANGE(0x00008000) CLOSE(0x80000000)	Project_A505.xlsx	Normal (0x00000080)	83939	2	Users₩YUN₩Desktop₩BMX_₩개인
13247584	2017-11-27 16:35:10	BASIC_INFO_CHANGE(0x00008000)	Project_A505.xlsx	Normal (0x00000080)	83939	2	Users₩YUN₩Desktop₩BMX_₩개인
13247680	2017-11-27 16:35:10	BASIC_INFO_CHANGE(0x00008000) CLOSE(0x80000000)	Project_A505.xlsx	Normal (0x00000080)	83939	2	Users₩YUN₩Desktop₩BMX_₩개인
13247776	2017-11-27 16:35:10	BASIC_INFO_CHANGE(0x00008000)	Project_A505.xlsx	Archive (0x00000020)	83939	2	Users₩YUN₩Desktop₩BMX_₩개인
13247872	2017-11-27 16:35:10	BASIC_INFO_CHANGE(0x00008000) CLOSE(0x80000000)	Project_A505.xlsx	Archive (0x00000020)	83939	2	Users₩YUN₩Desktop₩BMX_₩개인
13249120	2017-11-27 16:35:14	BASIC_INFO_CHANGE(0x00008000)	Project_A505.xlsx	Normal (0x00000080)	83939	2	Users₩YUN₩Desktop₩BMX_₩개인
13249216	2017-11-27 16:35:14	BASIC_INFO_CHANGE(0x00008000) CLOSE(0x80000000)	Project_A505.xlsx	Normal (0x00000080)	83939	2	Users₩YUN₩Desktop₩BMX_₩개인
13249312	2017-11-27 16:35:14	BASIC_INFO_CHANGE(0x00008000)	Project_A505.xlsx	Archive (0x00000020)	83939	2	Users₩YUN₩Desktop₩BMX_₩개인
13249408	2017-11-27 16:35:14	BASIC_INFO_CHANGE(0x00008000) CLOSE(0x80000000)	Project_A505.xlsx	Archive (0x00000020)	83939	2	Users₩YUN₩Desktop₩BMX_₩개인

③ 17:01	13367440	2017-11-27 17:01:24	DATA_EXTEND(0x00000002)	Project_A505.xlsx	Archive (0x00000020)	83939	2	Users₩YUN₩Desktop₩BMX_₩개인
	13367536	2017-11-27 17:01:24	DATA_OVERWRITE(0x00000001) DATA_EXTEND(0x00000002)	Project_A505.xlsx	Archive (0x00000020)	83939	2	Users₩YUN₩Desktop₩BMX_₩개인
	13367632	2017-11-27 17:01:24	DATA_OVERWRITE(0x00000001) DATA_EXTEND(0x00000002) DATA_TRUNCATION(0x00000004)	Project_A505.xlsx	Archive (0x00000020)	83939	2	Users₩YUN₩Desktop₩BMX_₩개인
	13367728	2017-11-27 17:01:24	DATA_OVERWRITE(0x00000001) DATA_EXTEND(0x00000002) DATA_TRUNCATION(0x00000004) CLOSE(0x80000000)	Project_A505.xlsx	Archive (0x00000020)	83939	2	Users₩YUN₩Desktop₩BMX_₩개인
	13367824	2017-11-27 17:01:24	A user changed the FILE_ATTRIBUTE_NOT_CONTENT_INDEXED attributeINDEXABLE_CHANGE(0x00004000) BASIC_INFO_CHANGE(0x00008000)	Project_A505.xlsx	Not Content Indexed (0x00002000)	83939	2	Users₩YUN₩Desktop₩BMX_₩개인
	13367920	2017-11-27 17:01:24	A user changed the FILE_ATTRIBUTE_NOT_CONTENT_INDEXED attributeINDEXABLE_CHANGE(0x00004000) BASIC_INFO_CHANGE(0x00008000) CLOSE(0x80000000)	Project_A505.xlsx	Not Content Indexed (0x00002000)	83939	2	Users₩YUN₩Desktop₩BMX_₩개인
	~중략~							
	13368016	2017-11-27 17:01:24	BASIC_INFO_CHANGE(0x00008000)	Project_A505.xlsx	Not Content Indexed (0x00002000)	83939	2	Users₩YUN₩Desktop₩BMX_₩개인
	13368112	2017-11-27 17:01:24	BASIC_INFO_CHANGE(0x00008000) CLOSE(0x80000000)	Project_A505.xlsx	Not Content Indexed (0x00002000)	83939	2	Users₩YUN₩Desktop₩BMX_₩개인
	13368208	2017-11-27 17:01:24	RENAME_OLD_NAME(0x00001000)	Project_A505.xlsx	Not Content Indexed (0x00002000)	83939	2	Users₩YUN₩Desktop₩BMX_₩개인
	13368304	2017-11-27 17:01:24	RENAME_NEW_NAME(0x00002000)	{0q{P1WQpfeMVb2ZM	Not Content Indexed (0x00002000)	83939	2	Users₩YUN₩Desktop₩BMX_₩개인
	13368400	2017-11-27 17:01:24	RENAME_NEW_NAME(0x00002000) CLOSE(0x80000000)	{0q{P1WQpfeMVb2ZM	Archive (0x00000020), Not Content Indexed	83939	2	Users₩YUN₩Desktop₩BMX_₩개인

	~중략~							
	13368688	2017-11-27 17:01:24	RENAME_OLD_NAME(0x00001000)	{0q{P1WQpfeMVb2ZM	Archive (0x00000020), Not Content Indexed	83939	2	Users₩YUN₩Desktop₩BMX_₩개인
	13368784	2017-11-27 17:01:24	RENAME_NEW_NAME(0x00002000)	DIWr_`6MyUZsnvRUp	Archive (0x00000020), Not Content Indexed	83939	2	Users₩YUN₩Desktop₩BMX_₩개인
	~중략~							
	13369168	2017-11-27 17:01:24	RENAME_OLD_NAME(0x00001000)	DIWr_`6MyUZsnvRUp	Archive (0x00000020), Not Content Indexed	83939	2	Users₩YUN₩Desktop₩BMX_₩개인
	13369344	2017-11-27 17:01:24	RENAME_NEW_NAME(0x00002000)	!EtiAdE d=z20xug1	Archive (0x00000020), Not Content Indexed	83939	2	Users₩YUN₩Desktop₩BMX_₩개인
	~중략~							
	13369728	2017-11-27 17:01:24	RENAME_OLD_NAME(0x00001000)	!EtiAdE d=z20xug1	Archive (0x00000020), Not Content Indexed	83939	2	Users₩YUN₩Desktop₩BMX_₩개인
	13369824	2017-11-27 17:01:24	RENAME_NEW_NAME(0x00002000)	sN5lUkhfx,u8Kc+b4	Archive (0x00000020), Not Content Indexed	83939	2	Users₩YUN₩Desktop₩BMX_₩개인
	~중략~							
	13370208	2017-11-27 17:01:24	RENAME_OLD_NAME(0x00001000)	sN5lUkhfx,u8Kc+b4	Archive (0x00000020), Not Content Indexed	83939	2	Users₩YUN₩Desktop₩BMX_₩개인
	13370304	2017-11-27 17:01:24	RENAME_NEW_NAME(0x00002000)	78Q0tYq9dccN4cCe7	Archive (0x00000020), Not Content Indexed	83939	2	Users₩YUN₩Desktop₩BMX_₩개인

	~중략~							
	13370688	2017-11-27 17:01:24	RENAME_OLD_NAME(0x00001000)	78Q0tYq9dccN4cCe7	Archive (0x00000020), Not Content Indexed	83939	2	Users₩YUN₩Desktop₩BMX_₩개인
	13370784	2017-11-27 17:01:24	RENAME_NEW_NAME(0x00002000)	X'12aUVZ1)gF-wksu	Archive (0x00000020), Not Content Indexed	83939	2	Users₩YUN₩Desktop₩BMX_₩개인
	~중략~							
	13371168	2017-11-27 17:01:24	RENAME_OLD_NAME(0x00001000)	X'12aUVZ1)gF-wksu	Archive (0x00000020), Not Content Indexed	83939	2	Users₩YUN₩Desktop₩BMX_₩개인
	13371264	2017-11-27 17:01:24	RENAME_NEW_NAME(0x00002000)	)Cz(PiLCdfz{uIAG!	Archive (0x00000020), Not Content Indexed	83939	2	Users₩YUN₩Desktop₩BMX_₩개인
	~중략~							
	13371456	2017-11-27 17:01:24	FILE_DELETE(0x00000200) CLOSE(0x80000000)	)Cz(PiLCdfz{uIAG!	Archive (0x00000020), Not Content Indexed	83939	2	Users₩YUN₩Desktop₩BMX_₩개인
④ 17:02	13375688	2017-11-27 17:02:10	FILE_CREATE(0x00000100)	Report.wer	Archive (0x00000020), Compressed (0x00000800),	83939	3	ProgramData₩Microsoft₩Windows₩WER₩ReportQueue₩NonCritical_IPX Assertion_b991e9b37c61384f
	~중략~							

[그림6-19] MFT Entry 번호 '83939'를 사용하는 $UsnJrnl 레코드

시간 순서대로 해당 MFT Entry의 변화를 살펴보면, 16:08 경에는 'BMX-S500J.xlsx'에서 'Project_A505.xlsx'로 파일명이 변경되었고, 16:35에는 'Project_A505.xlsx'의 속성값이 변경되었음이 확인된다. 17:01에는 총 7번 파일명이 변경되는 것을 확인할 수 있는데, 보통 임의의 문자로 변경되는 경우는 파일 삭제 프로그램을 이용하는 경우라고 유추해 볼 수 있다. 해당 MFT Entry가 최종적으로 할당하고 있는 파일은 'Report.wer' 파일로, 해당 경로에 접근해 보면 현재 해당 MFT Entry는 사용 중인 것이 확인된다.

위의 16:35 경 'Project_A505.xlsx'의 속성 값이 변경되는 것과 관련하여 해당 USN 번호 '13247392'를 기준으로 전후의 레코드를 살펴보면 변경 원인을 찾을 수 있다. 이전 USN 번호 '13242184'를 포함하여 40개의 레코드가 'NewFileTime.ini' 파일에 대한 정보를 기록하고 있고 바로 이어지는 USN 번호 '13246112'를 포함한 3개의 레코드는 'NEWFILETIME_X64.EXE-A814CD50.pf' 프리패치 실행을 기록하고 있다. 하지만 사용자가 'NewFileTime' 프로그램을 사용한 것이 확인되는데, 파일의 생성, 수정, 접근 MFT Entry 변경 시간 중 어떤 시간을 변경하였는지에 대해서는 확인되지 않는다.

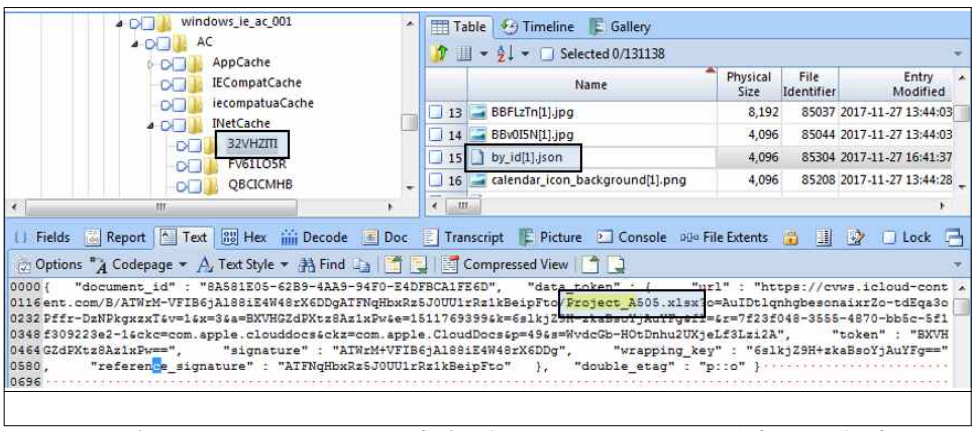
17:01 경의 7번의 파일명 변경 이벤트도 위와 마찬가지로 USN 번호 '13367440'을 기준으로 전후의 레코드를 살펴보면 그 원인을 알 수 있다. 이전 기록 레코드 USN '13365688'에서 파일삭제 프로그램인 'Eraser'가 실행되기 전 프리패치 파일 'ERASER.EXE-CE61944A.pf'이 실행된 것으로 보아 사용자가 'Eraser' 프로그램을 사용하여 'Project_A505.xlsx''을 삭제한 것을 알 수 있다.

③ 파일 업로드 흔적

16:40 경의 레코드에는 iCloud에 접속한 흔적이 기록되어 있다. 또한 16:41:21에는 파일이 업로드('singleFileUpload[1]')된 것이 확인된다. 하지만 업로드된 파일이 어떤 파일인지는 확인되지 않는다. 'singleFileUpload[1]' 레코드 이후로 확장자명 '.json'을 사용하는 파일들과 'Project_

A505.xlsx'과 유사한 이름의 파일들이 연속적으로 생성되고 삭제되는 것을 확인할 수 있다. 이러한 파일들의 변화는 'Users\YUN\AppData\Local\Packages\windows_ie_ac_001\AC\INetCache\~' 경로와 'Users\YUN\AppData\Local\Microsoft\Windows\INetCache\IE\~' 경로에서 이루어지고 있어 해당 경로를 분석한 결과 업로드한 파일 정보를 알 수 있었다.

확장자 '.json' 파일은 인터넷에서 자료를 주고받을 때 그 자료를 표현하는 방법으로 알려져 있다.[41] 경로들 중에서 우선 'Users\YUN\AppData\Local\Packages\windows_ie_ac_001\AC\INetCache\~'경로 내의 '.json' 파일들을 중점으로 내용을 검색해 본 결과 'by_id[1].json'파일에서 'Project_A505.xlsx'가 업로드 된 흔적이 발견되었다.



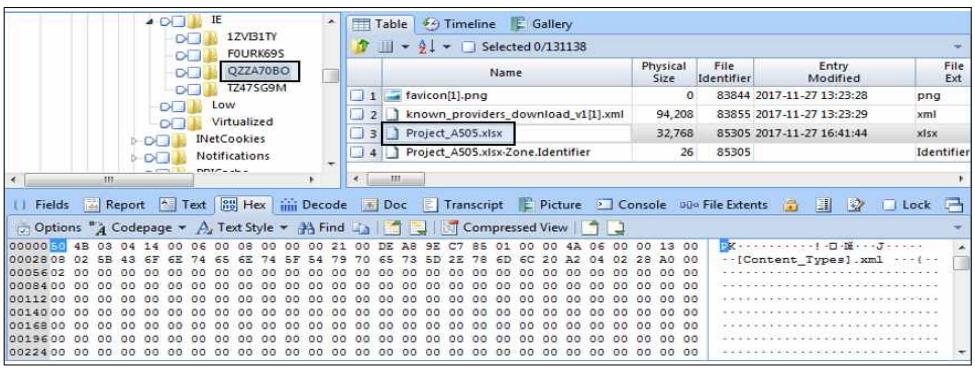
[그림6-20] 'by_id[1].json' 파일 내 'Project_A505.xlsx'업로드 흔적

한편 'Users\YUN\AppData\Local\Microsoft\Windows\INetCache\IE\~' 경로에서는 원본 파일과 동일한 'Project_A505.xlsx'가 발견되었는데, 위 경로 하위에 존재하는 디렉토리 'QZZA70BO' 내에서 발견되었다. 발견된

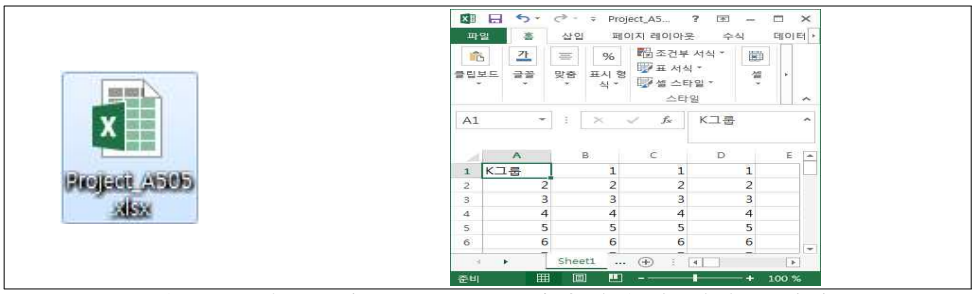
41) 속성-값 쌍으로 이루어진 데이터 오브젝트를 전달하기 위해 인간이 읽을 수 있는 텍스트를 사용하는 개방형 표준 포맷이다. 비동기 브라우저/서버 통신 (AJAX)을 위해, 넓게는 XML(AJAX가 사용)을 대체하는 주요 데이터 포맷이다. 특히, 인터넷에서 자료를 주고 받을 때 그 자료를 표현하는 방법으로 알려져 있다., 위키백과, https://ko.wikipedia.org/wiki/JSONhttps://ko.wikipedia.org/wiki/JSON, 2017.12.01.

엑셀 파일을 추출하면 그 내용도 확인할 수 있다. 이번 사례에서 'Project_A505.xlsx' 파일은 파일 삭제 프로그램에 의해 삭제되었다. 따라서 원본이 존재했던 경로에서는 해당 파일이 발견되지 않으나 iCloud로 업로드 하는 과정에서 생성된 캐시 기록들에서 원본 파일과 동일한 파일을 발견할 수 있었다.

키워드 검색을 통해서도 해당 파일들은 발견될 수 있을 것으로 보인다. 하지만 이렇게 발견된 파일들은 단편적으로 존재하므로 일련의 행위 순서와 관련짓지 못할 경우 특정 행위 혹은 사건과의 직접적인 관련성을 증명하기 어렵고 발견된 파일이 존재하는 경로에 어떻게 존재하게 되었는지에 대한 규명이 어려울 것으로 판단된다.



[그림6-21] iCloud로 업로드 하는 과정에서 생성된 'Project_A505.xlsx'파일



[그림6-22] 디렉토리 'QZZA70BO'에서 추출한 캐시로 저장된 'Project_A505.xlsx' 파일

3. 소결

두 개의 가상 시나리오 실험을 통해 $UsnJrnl 파일에 기록되는 정보가 특정 행위 시간을 기록하고 있음을 확인하였고, 이 레코드가 기록하는 파일들의 여러 정보를 통해 어떤 방법으로 추가 분석을 진행할 수 있는지를 보여 주었다.

시간 정보는 혐의를 입증하는데 중요한 요소로 작용하고 대다수의 분석 의뢰 사건들도 대부분 시간 특정에 대한 요청이다. 특히 중요 파일이 삭제되어 확인이 불가능한 경우 수사팀은 해당 파일에 대한 복원 요청과 함께 해당 파일을 삭제한 시간 특정을 요청한다. 이것은 사용자의 삭제 행위의 고의 또는 사건 관련성을 규명하기 위함일 것이다.

실제 위 증거인멸 사례 시나리오의 모티브가 된 사건 항소심에서 쟁점이 되었던 부분이 삭제된 파일의 삭제 시점과 대상 특정 문제였다. 피고인측에서는 특정 파일에 대한 삭제 시간을 알 수 없어 결과적으로 피고인이 인멸한 증거가 없다는 것과, 피고인이 인멸하였다는 파일의 이름과 내용 등이 특정되지 않아 그 파일이 증거에 해당하는 것인지도 확인할 수 없으며, 컴퓨터에서 이미 출력되어 압수된 파일과 피고인이 인멸한 증거의 동일성 등을 확인할 수도 없어 피고인의 방어권 행사에 심각한 지장을 초래하므로, 증거인멸죄가 성립할 수 없다는 것이었다.

하지만 해당 재판부는 피고인의 증거인멸 혐의를 인정하면서 다음과 같이 판단하였다. '피고인이 'Moo0' 프로그램의 항목 중 '5. 파일 이름 흔적-(MFT 내에 회복가능한 파일)'을 선택하여 실행함으로써 그 실행된 MFT 영역에서의 모든 파일에 대한 파일명, 파일크기, 작성일자, 사용권한, 데이터의 물리적 위치, 시간 정보, 삭제 상태, 파일 내용 참조 정보 등 파일에 관한 정보가 삭제되고 그 삭제된 공간에 아무런 정보가 없는 파일이 덮어씌워지게 되었다. (중략) MFT 영역이 삭제되면, 그 영역에서의 파일의 시간정보, 파일이름 등 중요한 디지털증거의 '정보'에 대한 복구가 사실상 불가능하게 된다고 봄이 상당하다. (중략) 다만 검찰은 실행되던 'Moo0' 프로그램이 피고인에 의하여 중단됨으로써 파일 정보가

삭제되지 않은 MFT 영역을 통해 삭제된 일부 파일들을 복구하였는데, 거기에는 이 사건 관련사건과 관계된 다수의 문건들이 존재한다. (중략) 피고인의 Moo0 프로그램의 실행 사이의 시간적, 장소적, 심리적 연관성을 종합하면, Moo0 프로그램 실행으로 복구가 불가능하거나 곤란하게 된 디지털정보는 그 실행이 없었다면 포렌식 전문가에 의하여 수월하게 복구될 가능성이 매우 높았고, 또 복구된 파일이 이 사건과 관계되는 자료임에 비추어, 이는 증거인멸죄의 '증거'에 해당한다 할 것이다.'[42]

삭제 프로그램으로 인해 삭제된 파일의 경우 원 파일에 대한 복구가 어렵다. 비할당 영역에서의 파일 카빙을 통해 복구하더라도 복구된 파일이 MFT 영역의 파일 정보 사이의 연관성을 전혀 확인할 수 없으므로 행위자의 증거인멸 행위와의 연관성 또한 확인이 불가능하다.

이번 연구에서 가장 주목할 사안은 완전 삭제 프로그램에 의해 삭제된 파일에 대해서도 MFT Entry 번호로 연결되는 $UsnJrnl 레코드로 파일명을 특정하거나 그 시간정보를 확인할 수 있다는 가능성을 보여준 것이다. 또한 연속적으로 생성되는 레코드의 일련의 과정을 통해 인과관계도 분명히 할 수 있었다.

$UsnJrnl 파일에는 시스템에 의해 실행된 무수히 많은 프로세스의 흔적들도 모두 기록되어 있다. 실제 환경에서는 분석 환경과 달리 더 많은 시스템 흔적들이 혼재해 있을 것이다. 위의 증거인멸 사례의 실제 사건에서는 피고인이 일괄적으로 삭제한 파일이 760,000여 개의 파일에 달하였다고 기록되어 있다. 이런 경우라면 실제 $UsnJrnl파일에서 그 삭제 흔적을 찾기는 쉽지 않았을 것이다. 하지만 만약 위의 사례에서처럼 사용자의 행위나 그 시간 정보가 중요 쟁점이 된 경우라면 반드시 검토할 필요가 있다.

---

42) 서울중앙지방법원, 2014.11.28 선고, 2014노2134 판결

제 7 장 결 론

수사와 마찬가지로 분석 또한 작은 단서 하나로 시작된다. 중요한 단서가 발견됨으로써 분석관이 분석해야할 대상과 방향이 결정된다.

앞서 살펴본 바와 같이 $UsnJrnl 파일은 사용자의 일련의 행위를 추적하고 시간 정보를 확보하는데 유용하다. 특정 파일에 대한 직접적인 정보를 찾을 수 없다 해도 사용자의 일련의 행위를 통해 또는 사용한 프로그램을 분석하여 데이터로 남지 않는 디지털 정보를 파악할 수 있는 단서로 활용될 수 있다.

하나의 작은 단서를 찾기 위해 수많은 레코드들을 꼼꼼히 봐야 하는 수고가 필요하겠으나, 의미있는 정보를 찾아내기 위해 작은 단서 하나라도 놓칠 수 없는 것이 분석관의 심정일 것이다.

이번 연구에서는 $UsnJrnl에 대한 기본 개념과 실제 검찰의 디지털 증거분석 사례를 살펴보았으며 증거인멸과 기술유출 사건의 가상 시나리오를 통해 구체적인 분석 방법과 방향을 제시하였다. $UsnJrnl 파일을 중심으로 분석하고자 추가로 진행해야하는 분석들은 일부 생략하였다. 분석 결과 $UsnJrnl 파일을 통해 탐지되는 유의미한 단서들이 상당수 존재하는 것을 확인하였고 특히 완전 삭제 프로그램을 통한 안티포렌식의 경우 MFT Entry 번호를 기준으로 $UsnJrnl 레코드들을 연결시키면 삭제된 혹은 변경된 파일들에 대한 정보를 비교적 구체적으로 확인할 수 있었다. 또한 $UsnJrnl 레코드가 가지는 정보들을 통해 여러 경로에 분산되어 있는 관련 파일들을 일관성 있게 연관지을 수 있다는 것도 확인할 수 있었다.

본 연구는 단 두 가지의 가상 시나리오만을 설정하였고, Windows 7과 8의 시스템 환경에서만 진행되었다. 따라서 해당 분석 방법이 모든 사건과 동일하게 적용할 수는 없을 것이나, 이번 연구의 목적은 그 방향을 제안하는 것으로 다른 환경의, 다른 사건의 경우 이와 유사한 방법으로 접근을 시도하면 유용한 결과를 얻을 것으로 기대한다. $UsnJrnl 파일에 대한 연구가 부족함은 앞서 언급한 바 있다. 따라서 본 연구에서도

다양한 이벤트들의 레코드 정보를 해석하는데 어려움이 있었고 본 연구에서 미처 해석해 내지 못한 정보들도 다수 있을 것이라고 생각한다. 앞으로 더 다양하고 활발한 연구가 진행되기를 바라면서 동시에 본 연구를 통해 디지털포렌식 수사관들이 하나의 유용한 분석 방법으로 사용되기를 바란다.

참 고 문 헌

[1] Brian Carrier, *File System Forensic Analysis.* Addison Wesley, 2006

[2] 정준석, 정원용, 「임베디드 개발자를 위한 파일시스템의 원리와 실습」 한빛미디어, 2006

[3] Christopher Lees, *Determining removal of forensic artefacts using the USN change journal*, Digital Investigation vol.10, no.4, 2013

[4] Frank Uijtewaal, Jeroen van Prooijen, *UsnJrnl Parsing for File System History Project Repor*t, University of Amsterdam, 2016

[5] 김진국, "NTFS 소개", http://forensic-proof.com/archives/427, (2017.12.2.)

[6] 오정훈, "NTFS Log Tracker", FORENSIC INSIGHT ; DIGITAL FORENSICS COMMUNITY IN KOREA, 2013

[7] 오정훈, "Advanced $UsnJrnl Forensics", FORENSIC INSIGHT ; DIGITAL FORENSICS COMMUNITY IN KOREA, 2015

[8] 탁희성, 이상진, "디지털 증거분석 도구에 의한 증거수집절차 및 증거능력 확보방안" 한국형사정책연구원, 2006

[9] 손지영, 김주석, "디지털 증거의 증거능력 판단에 관한 연구", 대법원 사법정책연구원 연구총서 2015-08, 2015

[10] 김영기, "디지털 증거의 진정성립부인과 증거능력 부여 방안", 한국형사판례연구(19), 2011

[11] 이승무, "범행현장성에 따른 디지털 포렌식 절차 모델 개발", 서울대학교, 2017

[12] 이완규, "디지털 증거 압수수색과 관련성 개념의 해석", 법조 2013 통권 686호, 2013

[13] 이완규, "사인작성 컴퓨터문서의 진정성립 입증과 증거능력", 한국형사판례연구(16), 2008

[14] 권양섭, "판례에서 바라본 디지털 증거의 증거능력에 관한 고찰", 한국정보보호학회 vol.26 no.5, 2016

[15] 이상미, 정대희, "디지털증거 압수수색 절차에서의 '관련성'의 문제", 한국형사정책연구 제28권 제2호, 2015

[16] 서울중앙지방법원 2014.11.28.선고, 2014노2134 판결

[17] 서울중앙지방법원 2014.6.5.선고, 2013고단3139 판결

[별지1] 변경 원인 플래그(Reason Flag)

(https://msdn.microsoft.com/en-us/library/windows/desktop/aa365722(v=vs.85).aspx)

Value	Meaning
BASIC_INFO_CHANGE 0x00008000	A user has either changed one or more file or directory attributes (for example, the read-only, hidden, system, archive, or sparse attribute), or one or more time stamps.
CLOSE 0x80000000	The file or directory is closed.
COMPRESSION_CHANGE 0x00020000	The compression state of the file or directory is changed from or to compressed.
DATA_EXTEND 0x00000002	The file or directory is extended (added to).
DATA_OVERWRITE 0x00000001	The data in the file or directory is overwritten.
DATA_TRUNCATION 0x00000004	The file or directory is truncated.
EA_CHANGE 0x00000400	The user made a change to the extended attributes of a file or directory.
ENCRYPTION_CHANGE 0x00040000	The file or directory is encrypted or decrypted.
FILE_CREATE 0x00000100	The file or directory is created for the first time.
FILE_DELETE 0x00000200	The file or directory is deleted.
HARD_LINK_CHANGE 0x00010000	An NTFS file system hard link is added to or removed from the file or directory.
INDEXABLE_CHANGE 0x00004000	A user changes the FILE_ATTRIBUTE_NOT_CONTENT_INDEXED attribute.
INTEGRITY_CHANGE 0x00800000	A user changed the state of the FILE_ATTRIBUTE_INTEGRITY_STREAM attribute for the given stream.
NAMED_DATA_EXTEND 0x00000020	The one or more named data streams for a file are extended (added to).
NAMED_DATA_OVERWRITE 0x00000010	The data in one or more named data streams for a file is overwritten.
NAMED_DATA_TRUNCATION 0x00000040	The one or more named data streams for a file is truncated.
OBJECT_ID_CHANGE 0x00080000	The object identifier of a file or directory is changed.
RENAME_NEW_NAME 0x00002000	A file or directory is renamed, and the file name in the USN_RECORD_V2 structure is the new name.
RENAME_OLD_NAME 0x00001000	The file or directory is renamed, and the file name in the USN_RECORD_V2 structure is the previous name.
REPARSE_POINT_CHANGE 0x00100000	The reparse point that is contained in a file or directory is changed, or a reparse point is added to or deleted from a file or directory.
SECURITY_CHANGE 0x00000800	A change is made in the access rights to a file or directory.
STREAM_CHANGE 0x00200000	A named stream is added to or removed from a file, or a named stream is renamed.
TRANSACTED_CHANGE 0x00400000	The given stream is modified through a TxF transaction.

[별지2] 소스 정보(Source Information)

(https://msdn.microsoft.com/en-us/library/windows/desktop/aa365722.aspx)

Value	Meaning
USN_SOURCE_AUXILIARY_DATA 0x00000002	The operation adds a private data stream to a file or directory. An example might be a virus detector adding checksum information. As the virus detector modifies the item, the system generates USN records. USN_SOURCE_AUXILIARY_DATA indicates that the modifications did not change the application data.
USN_SOURCE_DATA_MANAGEMENT 0x00000001	The operation provides information about a change to the file or directory made by the operating system. A typical use is when the Remote Storage system moves data from external to local storage. Remote Storage is the hierarchical storage management software. Such a move usually at a minimum adds the USN_REASON_DATA_OVERWRITE flag to a USN record. However, the data has not changed from the user's point of view. By noting USN_SOURCE_DATA_MANAGEMENT in the SourceInfo member, you can determine that although a write operation is performed on the item, data has not changed.
USN_SOURCE_REPLICATION_MANAGEMENT 0x00000004	The operation is modifying a file to match the contents of the same file which exists in another member of the replica set.
USN_SOURCE_CLIENT_REPLICATION_MANAGEMENT 0x00000008	The operation is modifying a file on client systems to match the contents of the same file that exists in the cloud.

[별지3] 파일 속성 (File Attribute)

(https://msdn.microsoft.com/en-us/library/windows/desktop/gg258117)

Value	Meaning
FILE_ATTRIBUTE_ARCHIVE 32 (0x20)	A file or directory that is an archive file or directory. Applications typically use this attribute to mark files for backup or removal.
FILE_ATTRIBUTE_COMPRESSED 2048 (0x800)	A file or directory that is compressed. For a file, all of the data in the file is compressed. For a directory, compression is the default for newly created files and subdirectories.
FILE_ATTRIBUTE_DEVICE 64 (0x40)	This value is reserved for system use.

FILE_ATTRIBUTE_DIRECTORY 16 (0x10)	The handle that identifies a directory.
FILE_ATTRIBUTE_ENCRYPTED 16384 (0x4000)	A file or directory that is encrypted. For a file, all data streams in the file are encrypted. For a directory, encryption is the default for newly created files and subdirectories.
FILE_ATTRIBUTE_HIDDEN 2 (0x2)	The file or directory is hidden. It is not included in an ordinary directory listing.
FILE_ATTRIBUTE_INTEGRITY_STREAM 32768 (0x8000)	The directory or user data stream is configured with integrity (only supported on ReFS volumes). It is not included in an ordinary directory listing. The integrity setting persists with the file if it's renamed. If a file is copied the destination file will have integrity set if either the source file or destination directory have integrity set. **Windows Server 2008 R2, Windows 7, Windows Server 2008, Windows Vista, Windows Server 2003 and Windows XP** : This flag is not supported until Windows Server 2012.
FILE_ATTRIBUTE_NORMAL 128 (0x80)	A file that does not have other attributes set. This attribute is valid only when used alone.
FILE_ATTRIBUTE_NOT_CONTENT_INDEXED 8192 (0x2000)	The file or directory is not to be indexed by the content indexing service.
FILE_ATTRIBUTE_NO_SCRUB_DATA 131072 (0x20000)	The user data stream not to be read by the background data integrity scanner (AKA scrubber). When set on a directory it only provides inheritance. This flag is only supported on Storage Spaces and ReFS volumes. It is not included in an ordinary directory listing. **Windows Server 2008 R2, Windows 7, Windows Server 2008, Windows Vista, Windows Server 2003 and Windows XP** : This flag is not supported until Windows 8 and Windows Server 2012.
FILE_ATTRIBUTE_OFFLINE 4096 (0x1000)	The data of a file is not available immediately. This attribute indicates that the file data is physically moved to offline storage. This attribute is used by Remote Storage, which is the hierarchical storage management software. Applications should not arbitrarily change this attribute.
FILE_ATTRIBUTE_READONLY 1 (0x1)	A file that is read-only. Applications can read the file, but cannot write to it or delete it. This attribute is not honored on directories. For more information, see You cannot view or change the Read-only or the System attributes of folders in Windows Server 2003, in Windows XP, in Windows Vista or in Windows 7.
FILE_ATTRIBUTE_RECALL_ON_DATA_ACCESS 4194304 (0x400000)	When this attribute is set, it means that the file or directory is not fully present locally. For a file that means that not all of its data is on local storage (e.g. it may be sparse with some data still in remote storage). For a directory it means that some of the directory contents are being virtualized from another location. Reading the file / enumerating the directory will be more expensive than normal, e.g. it will cause at least some of the file/directory content to be fetched from a remote store. Only kernel-mode callers can set this bit.
FILE_ATTRIBUTE_RECALL_ON_OPEN 262144 (0x40000)	This attribute only appears in directory enumeration classes (FILE_DIRECTORY_INFORMATION, FILE_BOTH_DIR_INFORMATION, etc.). When this attribute is set, it means that the file or directory has no physical representation on the local system; the item is virtual. Opening the item will be more expensive than normal, e.g. it will cause at least some of it to be fetched from a remote store.

FILE_ATTRIBUTE_REPARSE_POINT 1024 (0x400)	A file or directory that has an associated reparse point, or a file that is a symbolic link.
FILE_ATTRIBUTE_SPARSE_FILE 512 (0x200)	A file that is a sparse file.
FILE_ATTRIBUTE_SYSTEM 4 (0x4)	A file or directory that the operating system uses a part of, or uses exclusively
FILE_ATTRIBUTE_TEMPORARY 256 (0x100)	A file that is being used for temporary storage. File systems avoid writing data back to mass storage if sufficient cache memory is available, because typically, an application deletes a temporary file after the handle is closed. In that scenario, the system can entirely avoid writing the data. Otherwise, the data is written after the handle is closed.
FILE_ATTRIBUTE_VIRTUAL 65536 (0x10000)	This value is reserved for system use.

Abstract

Anyone who is a digital forensic investigator will have a lot of trouble to identify the crime. Depending on the type of incident, the behavior pattern of the user, or the issue of the incident, the subject of seizure is changed and the direction of analysis for the seizure is determined. For example, in the case of a technology leakages, the analysis begins by identifying whether a document related to the allegation exists in a particular device, or by identifying the route through which the technology has been leaked. If the evidence is an event of destruction, It begins with a grasp of how, when, and how it was deleted, or how to find and recover deleted documents.

As digital technology becomes more common and popular, it becomes more closely related with the daily life of the individual and the meaning of digital information as evidence is gradually increasing. On the other hand, as these technologies become more sophisticated and the access to new technical knowledge becomes easier, the knowledge of individuals' digital information becomes more intelligent, making it difficult for individual investigators to access digital information. In particular, as the privacy of individuals or the security of the enterprise is strengthened, anti-forensic technology becomes more diverse and more precise, so digital forensic investigators who analyze these digital evidence and develop the results need more variety and depth of technology and information Learning is required.

Most of the requests for analysis of digital evidence are time information about a user's specific action. In particular, it is information about the execution time of an action that is an issue, for example, the time when a specific file exists, a time when the file is created, changed, or deleted. In all cases, time information is an

essential element of criminal activity, and digital evidence is no exception. From the viewpoint of digital forensics, there are many factors that can confirm such information, but the $UsnJrnl file discussed in this study records the behavior of users and systems with relatively specific time information due to the nature of the function. However, there is a lack of research on the $UsnJrnl file that can be utilized by digital forensic investigators at home and abroad. The study of various investigation techniques is useful for presenting methods of approaching case to practitioners and solving the case. Through this study, it is possible to utilize $UsnJrnl file which is easy to track user's behavior based on time information. I propose a method of analysis.

In this study, I showed the case of using information detected from $UsnJrnl file in real work, and presented a concrete analysis method by constructing a hypothetical scenario. In this case, I can confirm the deletion time information of the file which is the issue of the case. I show through this experiment that we can track the trace even for the complete deletion program or other anti forensic activities.

$UsnJrnl, it's not just that it's easy to find the information you need from the myriad of information that a file may contain. But if there is only one percent chance, it will not be easily overlooked. I hope that this study will be useful for digital forensic investigators while at the same time feeling the need for more diverse and active research.