Journal of The Korea Institute of Information Security & Cryptology ISSN 1598-3986(Print)
VOL.25, NO.5, Oct. 2015 ISSN 2288-2715(Online)
http://dx.doi.org/10.13089/JKIISC.2015.25.5.1143

ESE 데이터베이스 내의 삭제된 레코드 복구 기법*

김 정 현,† 최 종 현, 이 상 진‡
고려대학교 정보보호대학원

A recovery method for deleted records in the ESE Database*

Jeong-hyeon Kim,† Jong-hyun-Choi, Sang-jin Lee‡
Center for Information Security Technologies, Korea University

요 약

Extensible Storage Engine (ESE) 데이터베이스는 Microsoft가 개발한 데이터베이스로 Internet Explorer, Spartan과 같은 웹브라우저와 Windows Search, System Resource Usage Monitor와 같은 윈도우 시스템에서 주요하게 사용된다. 기존의 ESE 데이터베이스 뷰어는 잘못된 값을 출력할 수 있고 수집 환경과 파일의 상태에 따라 파일을 읽지 못하는 경우가 존재한다. 또한 삭제된 레코드 복구 도구는 일부 프로그램에 한정적이고 모든 테이블을 복구하지 못한다. 본 논문에서는 ESE 데이터베이스 구조를 분석하여 개별 데이터베이스가 아닌 범용적으로 적용할 수 있는 삭제된 레코드의 복구 기법을 제안하며 이를 도구로 구현하여 실험한 결과를 제시한다.

ABSTRACT

Extensible Storage Engine (ESE) database is a database developed by Microsoft. This database is used in web browser like Internet Explorer, Spartan and in Windows system with Windows Search, System Resource Usage Monitor. Previous ESE database viewer can display an incorrect result and can't read the file depending on collected environment and status of files. And the deleted record recovery tool is limited to some program and cannot recover all tables. This paper suggests the universal recovery method for deleted records and presents the experimental results through development of tool.

Keywords: ESE database analysis, ESE database format, ESE database forensic

I. 서 론

Microsoft사가 개발한 Extensible Storage Engine(ESE) 또는 JET Blue라고 알려진 데이터베이스 엔진은 1994년부터 사용되기 시작했고 Internet Explorer, Spartan과 같은 웹 브라우저와 Windows Search, System Resource Usage Monitor와 같은 윈도우 시스템에서 주요하게 사용된다[1]. ESE 데이터베이스는 Windows OS에서 시스템 및 사용자의 주요 기록들을 저장, 관리하는 목적으로 사용되기 때문에 이에 대한 연구는 포렌식 관점에서 중요하다.

Metz는 ESE 데이터베이스의 파일 포맷을 분석하였으나 일부 파악하지 못한 영역이 다수 있어 이 문서만으로 ESE 데이터베이스를 파악하는 것은 매우 어렵다[2].

ESE 데이터베이스 뷰어는 EseDbViewer, EseDatabaseView가 존재하며 모두 문제점이 존재하며 삭제된 레코드는 열람하지 못한다. EseDbViewer는 데이터베이스 API를 사용했기 때문에 정확한 결과를 출력하나 정상 종료된 파일만 읽을 수 있다[3]. EseDatabaseView는 테이블이 복

접수일(2015년 6월 23일), 수정일(2015년 9월 11일), 게재확정일(2015년 9월 14일)
* 이 논문은 2015년도 정부(미래창조과학부)의 재원으로 한국연구재단-공공복지안전사업의 지원을 받아 수행된 연구임(2012M3A2A1051106)
† 주저자, americano@korea.ac.kr
‡ 교신저자, sangjin@korea.ac.kr(Corresponding author)

잡한 데이터 구조로 되어 있을 경우 잘못된 값을 출력하거나 빈 화면을 출력한다[4]. 이는 데이터베이스 포맷을 제대로 파악하지 못한 결과로 보인다.

이 외에도 Windows Search와 Internet Explorer가 사용하는 ESE 데이터베이스 파일 복구에 관한 연구가 진행되었으나 이 방법은 특정 파일, 일부 테이블에만 적용되고 삭제된 레코드를 완벽하게 복구하지 못한다[5][6].

본 논문에서는 ESE 데이터베이스 구조를 분석하여 레코드의 저장 방식 및 삭제 변화를 조사했다. 그리고 정상 레코드 추출 및 삭제된 레코드 복구 기법을 제시하고 이를 도구로 구현했다.

2장에서는 ESE 데이터베이스 구조 분석과 삭제된 레코드 복구와 관련된 연구를 소개하고, 3장에서 ESE 데이터베이스 구조를 상세히 설명한다. 4장에서는 레코드 삭제 후 데이터베이스 구조 및 내부 변화를 관찰하고, 5장에서는 4장에서 관찰한 정보를 근거로 삭제된 레코드 복구 기법을 제시한다. 6장에서는 제시한 복구 방법을 실제 구현 후 실험결과에 의한 성능을 평가한다. 마지막으로 7장에서는 결론 및 향후 연구 방향을 기술한다.

1.1 용어 설명

본 논문에서는 Metz[2]에서 사용한 용어를 동일하게 사용했다. Fig. 1.은 ESE 데이터베이스의 전체적인 구조를 나타낸 그림이다.

"Item"은 데이터베이스 파일에서 의미를 가지면서 특정 오프셋에 위치하는 가장 작은 단위를 말한다. "Data"는 Item이 가지고 있는 실제 값을 의미한다. "Record"는 데이터베이스의 레코드 또는 행을 의미한다. "Tag"는 단일 Record의 오프셋과 크기를 가지고 있는 Item을 의미한다. "Page"는 ESE 데이터베이스가 Record들을 저장 및 관리하기 위해 한 개 단위로 취급하는 레코드의 집합이다.

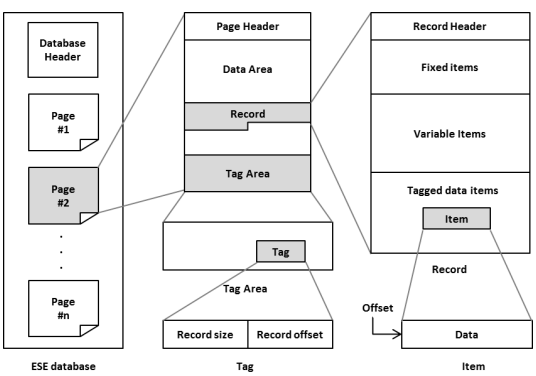

Fig. 1. Overall structure of ESE database

데이터베이스는 크게 2가지 상태로 나뉘게 되는데 응용프로그램이 데이터베이스를 사용하는 중의 상태를 "Dirty" 상태라 하고 데이터베이스 API를 사용하여 트랜잭션을 모두 처리한 상태를 "Clean" 상태라 한다.

II. 관련 연구

2007년 Microsoft는 ESE에 대한 문서를 공개하였고 2013년에 마지막으로 업데이트하였다[1]. Gordan(2009)의 연구에서는 ESE 데이터베이스 레코드를 저장하기 위해 사용되는 데이터 포맷의 개요를 포함하여 Windows Search의 아티팩트를 조사하였다[7]. 또한, Douglas는 Microsoft 데이터베이스 API를 사용하여 Desktop Search를 추출하는 방법에 대해 연구하였다[8].

Metz는 ESE 데이터베이스 파일 포맷을 분석하였고 Windows Search에서 사용되는 ESE 데이터베이스의 내부 구조와 스키마를 구체적으로 분석하였다[7]. 이후 역공학으로 ESE 데이터베이스 파일 포맷을 분석하였고 libesedb 프로젝트을 진행하여 라이브러리 파일을 개발했다[8]. 이 프로젝트는 현재에도 진행 중이며 최근에는 Windows Help and Support Services, Windows Mail, Windows Search, Windows Security, Windows Update 파일을 분석하고 있다.

H. Chivers와 C. Hargreaves는 Windows Search를 분석하여 B-Tree 구조로 인해 삭제된 레코드가 남는다는 사실을 발견하였고 레코드의 구조를 통해 삭제된 레코드를 카빙하는 방식을 제시하였다[5]. H. Chivers는 같은 방식을 사용해 Internet Explorer 10 이상 버전에서 사용되는 WebCacheV01.dat 파일에서 삭제된 레코드를 카빙한 뒤 복구된 레코드를 분석하여 type 항목으로부터 Private 브라우징을 판별할 수 있는 비트 마스크를 식별해냈다[6]. 또한 Private 브라우징이 어떤 상황에서 복구가 가능한지에 대해 연구하였다. 두 연구 모두 데이터베이스 API 기반으로 동작하는 도구를 사용하여 레코드를 분석하였다. 이 도구는 Dirty 상태의 데이터베이스를 Clean 상태로 바꾸

어야만 한다. 이는 Dirty 상태에서만 남는 가장 최근의 Private 브라우징 기록을 사라지게 만들 수 있다. 또한 이 도구는 Long value 페이지에 저장되는 값과 레코드의 마지막 항목을 복구하지 못한다.

본 논문에서는 위의 문제점을 해결할 수 있는 방안을 제시하고 도구를 구현해 범용적으로 ESE 데이터베이스의 삭제된 레코드를 복구할 수 있음을 입증한다.

III. ESE 데이터베이스 구조 분석

ESE 데이터베이스 구조의 대부분은 Metz에 의해 문서로 정리되었다[2]. 따라서 본 논문에서는 분석을 위해 꼭 필요하거나 추가적으로 분석한 내용만 작성한다. ESE 데이터베이스 파일은 데이터베이스 헤더를 제외한 다수의 페이지들로 구성되어 있고 B-tree 구조로 페이지들을 관리한다[1]. Fig.2.는 ESE 데이터베이스의 테이블 관리와 내부 구조를 도식화한 그림이다.

ESE 데이터베이스 내에는 다수의 테이블이 존재하고 테이블에 대한 정보는 MsysObject라고 하는 카탈로그 테이블에서 관리한다[2]. 특정 테이블은 경우에 따라 큰 데이터를 저장하기 위해 LV 라는 이름의 하위 테이블을 포함한다. 테이블은 Father data page(FDP) 번호라고 알려진 자기만의 식별번호가 있으며 이는 테이블에 속한 모든 페이지의 헤더에 명시되어 있다. 큰 데이터를 저장하기 위해 사용되는 LV라는 이름의 하위 테이블 또한 독자적인 FDP 번호를 사용하며 상위 테이블과 같은 방식으로 관리된다.

MsysObject 테이블은 일반적으로 4번째 페이지에 위치하며 자신을 포함한 모든 테이블들에 대한 정보를 설명한다. 이러한 정보는 ObjidTable, Type, Id, ColtypOrPgnoFDP, Space Usage, Name 등으로 구성되어 있다. ObjidTable은 자신이 속한 테이블의 FDP 번호이다. 이것을 제외한 다른 필드는 Type필드의 값에 따라 해석 방법이 달라진다. Table 1.은 Type 필드에 따른 레코드 해석 방법을 정리한 표이다.

페이지는 ESE 데이터베이스 내에서 레코드들을 저장 및 관리하기 위해 사용하는 논리적 단위이며, 순차적으로 헤더, 데이터 그리고 태그 영역으로 구성된다.

지금까지 알려진 페이지 종류는 Data, Branch, Empty, Space tree, Index, Long value 가 있으며 페이지 헤더의 Page flags 값으로 페이지를 구분할 수 있다. 레코드 저장방식은 페이지 종류에 종속적이며 Data, Branch, Long value 페이지 분석을 통해 실제 저장된 레코드를 확인할 수 있다.

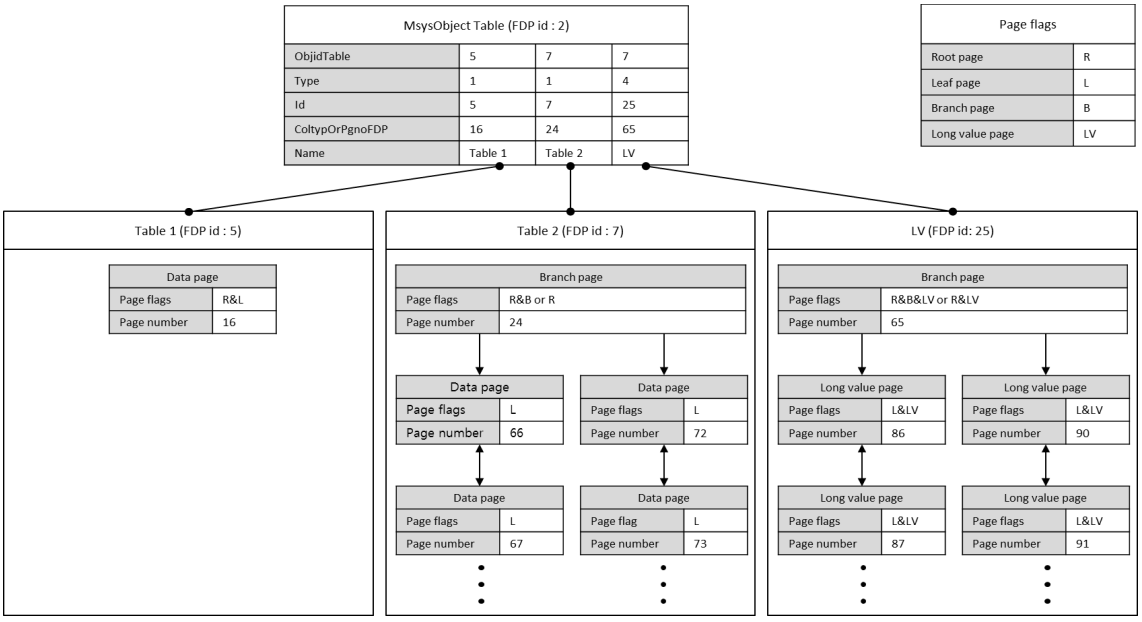

Fig. 2. Internal structure in ESE database

Table 1. Records type in MsysObject table

Type	Id	ColtypOrPgnoFDP	Space Usage	Name
1	FDP id	FDP number	number of pages	table name
2	column id	column type	column size	column name
4	LV FDP id	LV FDP number	number of pages	“LV”

Data 페이지란 테이블의 실제 데이터가 기록된 페이지를 말한다. Branch 페이지란 B-tree 구조에서 하위 레벨의 페이지의 번호가 기록된 페이지를 말한다. Long value 페이지는 LV 테이블에서 큰 데이터를 저장하기 위해 사용되는 페이지를 말한다.

IV. 레코드 삭제 실험

4.1 실험 환경

레코드 삭제 시 변경되는 정보를 확인하기 위해 Internet Explorer 10 이상의 버전에서 사용되는 WebCacheV01.dat 파일을 사용했다. 이 파일은 인터넷 사용 기록 및 캐쉬 파일 정보 등을 보관하기 위해 사용된다. 레코드 삭제를 위해 Internet Explorer 프로그램의 검색 기록 삭제 기능을 사용했고 삭제 가능한 모든 기록을 삭제했다. 이 파일은 일반 사용자가 약 1년 동안 사용된 컴퓨터에서 수집한 파일이고 수집하기 약 4개월 전에 검색 기록을 삭제한 내역이 있다.

4.2 실험 결과

Table 2.은 레코드 삭제 전과 삭제 후의 ESE 데이터베이스 구조에 대한 정보를 나타낸 표이다. 삭제 결과 전체 페이지 개수는 변함이 없으나 B-Tree 구조가 변하여 일부 Branch 페이지는 자식페이지가 없는 단일 Root 페이지로 바뀌었다. 특히 실제 데이터가 기록되었던 다량의 Leaf 페이지는 Empty 페이지로 변했다.

레코드 삭제 시 데이터베이스의 구조가 변함과 동시에 페이지의 내부도 변하게 된다. Fig.3.은 ESE 데이터베이스에서 모든 레코드가 삭제됐을 때 페이지 변화 결과를 나타낸 그림이다. 데이터영역과 태그영역은 삭제되지 않은 반면 레코드 개수를 나타내는 헤더 영역의 Available page tag 항목의 값이 변했다. 페이지가 사용되지 않을 경우 Page flags 항목에 Empty page 플래그가 설정되었다.

Table 2. Changed information of page flag by when all of records are deleted in WebCacheV01.dat

Page		Original	Deleted
Data	Root	37	41
	Branch	21	19
	Leaf	1271	469
	Empty	23108	23930
	Total	24437	24459
Long value	Root	25	27
	Branch	10	8
	Leaf	754	443
	Empty	39910	40200
	Total	40699	40678
Index	Root	14	14
	Branch	2	2
	Leaf	3	3
	Empty	6	6
	Total	25	25
Space Tree	Root	36	35
	Branch	2	2
	Leaf	4	4
	Empty	2	2
	Total	44	43
Total		65205	65205

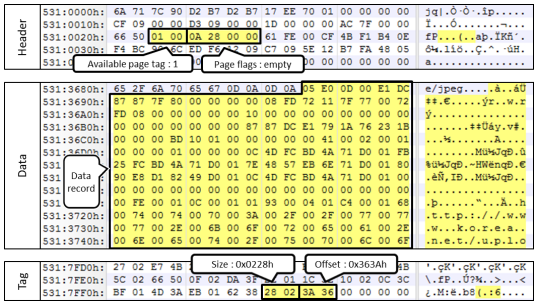

Fig. 3. Internal change inside the page when all the records have been deleted in the ESE database

V. 레코드 복구 기법 제시

ESE 데이터베이스 파일 내에는 2가지 이유 때문에 삭제된 레코드 복구가 가능하다. 첫 번째로 Data 페이지 또는 Long value 페이지가 Branch 페이지로 변한 경우 기존의 데이터는 Branch 페이

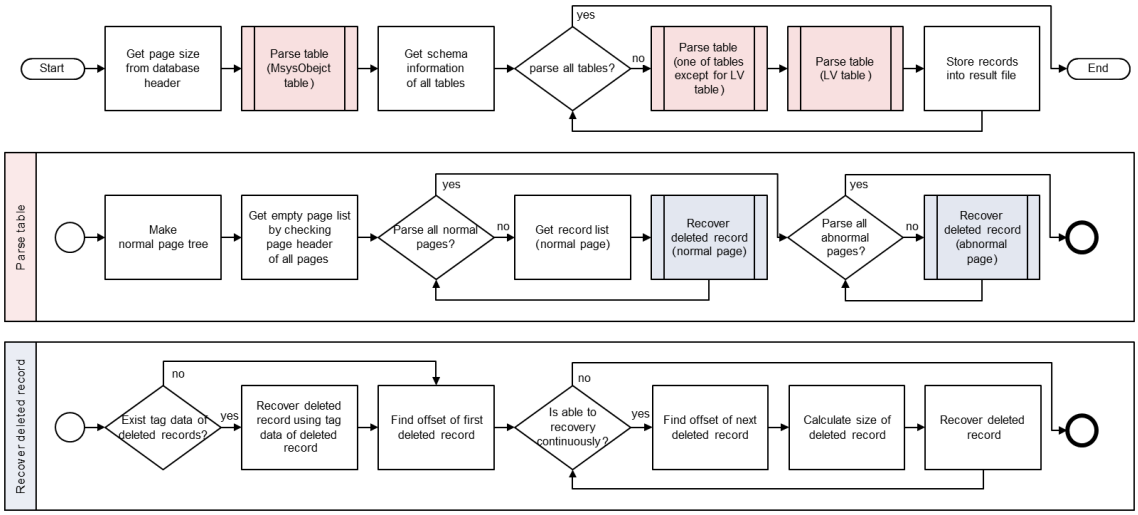

Fig. 4. Recovery procedure for deleted records in the ESE Database

지에 그대로 잔존한다. 두 번째는 레코드를 삭제하면 태그영역과 데이터영역은 삭제하지 않고 레코드 개수와 페이지 종류만 변경된다.

Fig.4.는 ESE 데이터베이스 파일의 삭제된 레코드 복구 절차를 표현한 순서도이다. ESE 데이터베이스 파일의 시작 부분에는 데이터베이스 헤더와 그것의 복사본이 존재한다. 데이터베이스 헤더에는 고유 시그니처, 페이지 크기, 파일 상태, 버전 등과 같은 기본적인 정보가 기록되어 있다. 버전과 페이지 크기에 의해 페이지 오프셋과 레코드 저장 방식이 결정된다.

레코드 복구를 위해선 해당 페이지가 속한 테이블의 스키마 정보가 필요하다. 카탈로그 테이블의 Data 페이지에는 삭제된 테이블과 관련된 레코드가 존재할 수 있다. 따라서 카탈로그 테이블의 삭제된 레코드를 복구하면 삭제된 테이블 스키마 정보를 알 수 있다.

ESE 데이터베이스 파일 내에는 더 이상 사용하지 않기 때문에 Branch 페이지에 기록되지 않은 페이지가 존재할 수 있다. 이러한 비정상적인 페이지의 헤더에는 기존에 속해 있었던 테이블의 번호가 기록되어 있기 때문에 이 테이블의 스키마 정보를 알고 있다면 삭제된 레코드의 복구가 가능하다. 따라서 정상적으로 사용 중인 페이지 외에 파일에 남아 있는 비정상 페이지까지 복구할 페이지 목록에 추가한다.

레코드의 크기는 가변적이기 때문에 예측할 수 없고 이 정보가 없으면 레코드를 정확히 복구하지 못한다. 하지만 삭제되지 않은 페이지의 태그 정보를 이용해 레코드 크기를 구할 수 있다. 이 밖에도 레코드가 공백 없이 연속되어 기록되는 특성을 이용해 다음 레코드의 시작위치로 레코드가 끝나는 지점의 오프셋을 찾을 수 있다.

삭제된 레코드의 시작 오프셋은 테이블 스키마 정보와 레코드 구조를 이용해 찾을 수 있다. Fig.5.는 Data 페이지 내에 있는 데이터 레코드의 구조를 도

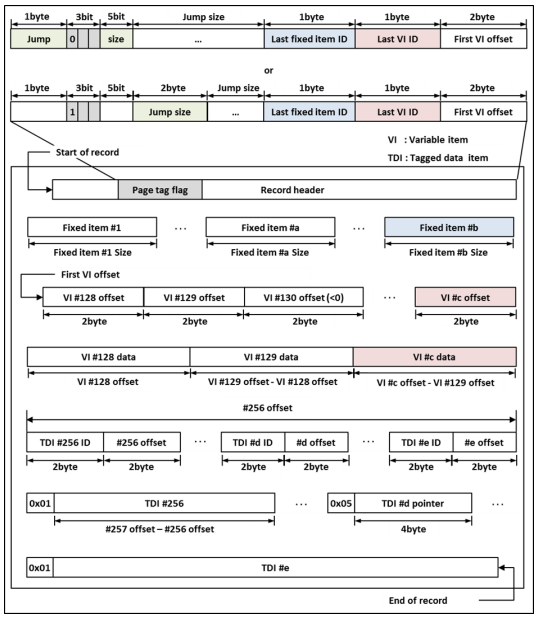

Fig. 5. Data record structure

식화한 그림이다. 데이터 레코드는 Record header, Last fixed item ID, Last variable item ID, First variable item Offset, Fixed items, Variable items, Tagged data items으로 구성되어 있다. 여기서 Record header, Last fixed item ID, Last variable item ID, First variable item Offset이 가질 수 있는 값의 범위를 이용해 삭제된 레코드의 시작 오프셋을 찾을 수 있다. Table 3.은 위의 항목을 이용해 삭제된 레코드의 시작 오프셋을 찾는 방법을 나타낸 표이다.

이 외에도 이미 알려진 테이블 정보를 이용해 레코드를 복구할 수 있다. 예로서 IE10, IE11, Spartan이 사용하는 WebCacheV01.dat 파일의 Container 테이블은 레코드 헤더 뒤에 4바이트가 0x117F7700h 값으로 고정적이고 동일한 구조로 레코드가 저장된다.

이 밖에 Branch page, Long value page의 구조와 분석 방법은 Appendix에서 상세히 설명한다.

Table 3. Method for finding start point of deleted records

Precedence	Conditional statements		
1	The first byte	!=	0
2	Recrod size, Offset	<	Page size
3	Jump size	<	The rest of the slack area
4	Last fixed item ID	In	Fixed item column
5	Last variable item ID	In	Variable item column
6	Last variable item Offset	<	The rest of the slack area
7	Last variable item Offset	<	Fixed item's data area

VI. 구현 및 성능 평가

앞에서 제시한 복구 방법을 검증하기 위해 데이터베이스 API를 사용하지 않고 ESE 데이터베이스 파일의 정상 레코드 추출 및 삭제된 레코드를 복구하는 도구를 구현했다.

구현한 도구의 정확성 여부를 평가하기 위해 이전에 연구되었고 다양한 종류의 ESE 데이터베이스의 파일들을 테스트 셋으로 사용했다. Table 4.는 이에 대한 정보와 수집환경을 설명한다.

정상 레코드 추출의 성능을 평가하기 위해 기존에 개발된 데이터베이스 API 기반 도구인 EseDbViewer 도구를 사용한 결과와 비교했다[4]. 그 결과 본 논문에서 구현한 도구와 레코드 개수와 내용이 동일하게 추출되었다.

삭제된 레코드 복구의 성능을 평가하기 위해 유일한 삭제 레코드 복구 도구인 ESECarve 도구를 사용한 결과와 비교했다. ESECarve 도구는 WebCacheV01.dat 파일과 Windows.edb 파일만 복구할 수 있고 일부 테이블만 복구가 가능하다[6]. 정확한 비교를 위해 중복된 항목은 제외하고 ESECarve가 복구한 테이블의 레코드만 비교했다. 그 결과 복구한 레코드의 개수는 동일했지만 실제 데이터를 확인하면 ESECarve는 Fig.6.과 같이 모든 레코드의 마지막 필드 항목과 LV테이블에 저장된 항목을 복구하지 못한 반면 본 논문에서 구현한 도구는 LV테이블에 저장된 항목 중 더 이상 존재하지 않는 경우를 제외하고 Fig.7.과 같이 정상적으로 레코드를 복구했다. ESECarve가 복구하지 못한 ResonseHeaders 필드에는 방문한 웹 사이트, 다운로드 경로, 캐쉬 파일 정보와 같은 중요한 정보가 저장된다. Fig.8.은 이 필드로부터 방문한 웹 사이트 정보를 추출해낸 결과이다.

WebCacheV01.dat 파일과 Windows.edb 파일을 제외한 아직 연구되지 않은 다른 종류의 ESE 데이터베이스 파일로 삭제된 레코드 복구 실험을 한 결과 SRUDB.dat을 제외한 다른 파일은 문제없이 삭제된 레코드를 복구할 수 있었지만 SRUDB.dat 파일은 삭제된 레코드를 복구할 수 없었다. 그 이유

Fig. 6. Results recovered by using ESE carve

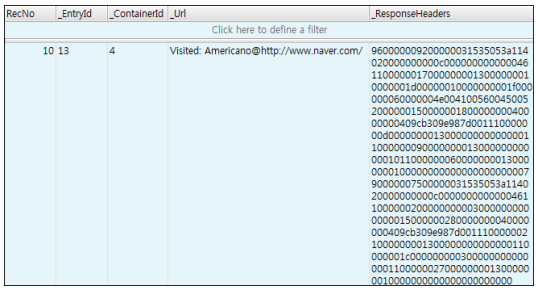

Fig. 7. Results recovered by using our recovery method

Table 4. Test set to measure performance

File name	OS version	Path	Program
WebCacheV01.dat	Windows 10	%LOCALAPPDATA%\Spartan\Database	Spartan
WebCacheV01.dat	Windows 7	%LOCALAPPDATA%\Microsoft\Windows\WebCache	Internet Explorer
Windows.edb	Windows 7	%PROGRAMDATA%\Microsoft\Search\Data\Applications\Windows	Windows Search
SRUDB.dat	Windows 8.1	%WINDIR%\system32\SRU	System resource usage monitor
DataStore.edb	Windows 7	%WINDIR%\SoftwareDistribution\DataStore	Windows Update
contacts.edb	Windows server 2008	%LOCALAPPDATA%\Microsoft\Windows Live\Contacts\[account name]\version\DBStore	Windows Live
WLCalendarStore.edb	Windows server 2008	%LOCALAPPDATA%\Microsoft\Windwos Live\Calendars\[account name]\DBStore	Windows Live
meta.edb	Windows 10	%LOCALAPPDATA%\Microsoft\Windows\Setting Sync\remotemetastore\v1	Windows store
meta.edb	Windows 10	%LOCALAPPDATA%\Microsoft\Windows\Setting Sync\metastore	Windows store

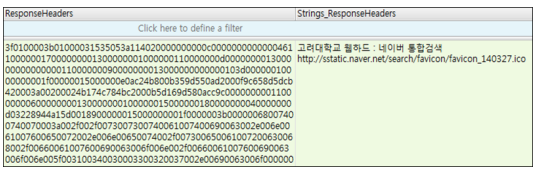

Fig. 8. Records of visited websites in ResponseHeader field

를 확인한 결과 삭제된 레코드는 모두 일정한 값으로 덮어쓰여져 있었다.

VII. 결 론

ESE 데이터베이스는 Windows OS에서 시스템 및 사용자의 주요 기록들을 저장, 관리하는 목적으로 사용되기 때문에 이에 대한 연구는 포렌식 관점에서 중요하다. ESE 데이터베이스 파일의 삭제된 레코드 복구에 관한 많은 연구가 진행되었지만 이전의 연구는 Internet Explorer와 Windows Search 프로그램에만 한정적이고 Dirty 상태의 파일은 복구하지 못하며 마지막 칼럼의 데이터를 복구하지 못하는 문제점이 존재한다. 본 논문에서 ESE 데이터베이스 파일의 구조를 분석하여 데이터베이스 API를 사용하지 않고 정상 레코드를 추출했고 그 결과를 데이터베이스 API를 사용한 것과 비교했다. 결과는 동일했고 이는 구조를 정확하게 분석했음을 보여준다. 또한 데이터베이스 구조를 기반으로 삭제된 레코드를 복구하는 기법을 연구했고 그 결과 기존 연구의 문제점을 해결하면서 다양한 버전의 파일로부터 삭제된 레코드를 복구할 수 있는 도구를 구현했다.

이 도구는 알려지지 않은 테이블의 레코드 구조에 대한 정보가 담겨 있는 레코드의 헤더가 손상될 경우 삭제된 레코드를 복구하지 못하는 한계점이 있다. 복구할 테이블의 스키마 정보를 알고 있다면 레코드 헤더 없이 삭제된 레코드를 복구할 수 있기 때문에 향후 Windows에서 사용하는 다양한 ESE 데이터베이스 파일을 조사하여 테이블 구조를 정리하고 손상된 레코드를 복구할 수 있는 방법에 대한 연구를 진행할 계획이다.

Appendix. A. Branch 페이지 분석 방법

페이지들은 B-Tree 구조로 관리되며 트리의 레벨이 증가하면 Data 페이지 및 Long Value 페이지는 Branch 페이지로 변한 뒤 하위 레벨의 페이지 번호를 기록하는 용도로 사용된다. Fig.9.은 Branch 페이지에서 데이터 영역에 존재하는 레코드의 구조를 나타낸 그림이다. 레코드에서 페이지 번호를 확인한 뒤 다음 식의 계산을 통해 실제 데이터가 저장된 페이지의 오프셋을 알 수 있다.

$$page\,offset = (page\,number + 1) \times page\,size$$

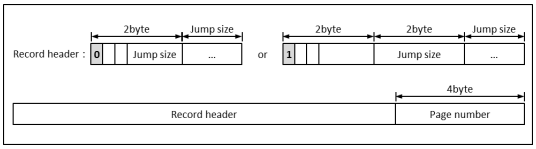

Fig. 9. Branch record structure

Appendix. B. Long value 페이지 분석 방법

특정 테이블은 큰 데이터를 레코드에 직접적으로 저장하지 않고 포인터를 사용해 LV라는 이름의 하위 테이블에 저장한다. LV 테이블은 Long value 페이지를 사용하며 여러 개의 레코드를 사용해 하나의 데이터를 저장한다. 데이터의 크기가 페이지 크기보다 클 경우 여러 개의 페이지에 연속해서 데이터를 저장한다.

Long value 페이지는 3 종류의 레코드 구조를 이용해 하나의 데이터를 저장한다. Fig.10.은 Long value 페이지에서 사용하는 레코드의 구조를 나타낸 그림이다. 그림의 첫 번째 상자는 Long value 페이지의 첫 번째 레코드 구조이다. 이 레코드에는 다음 레코드의 LV 번호와 이전 페이지에서 데이터를 얼마만큼 기록했는지에 대한 정보가 기록되어 있다. 두 번째 상자는 실제 데이터의 전체 크기를 기록하기 위해 사용되는 레코드 구조이다. 마지막으로 세 번째 상자는 실제 데이터를 기록하기 위해 사용되는 레코드 구조이다.

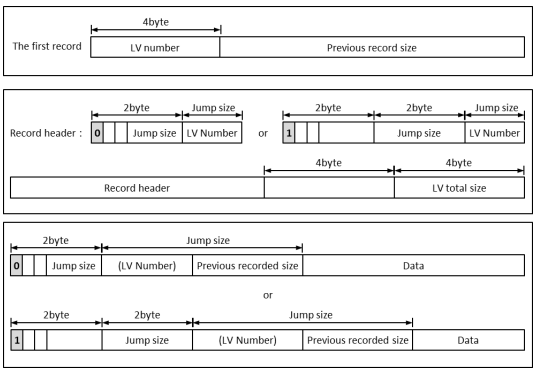

Fig. 10. Long value record structure

References

[1] Mircosoft, "Extensible Storage Engine," https://technet.microsoft.com/library/Cc961824

[2] J. Metz, "Extensible Storage Engine (ESE) Database File (EDB) format specification," https://github.com/libyal/libesedb/tree/master/documentation

[3] woanware, "EseDbViewer v1.0.6," http://www.woanware.co.uk/forensics/esedbviewer.html

[4] NirSoft, "ESEDatabaseView v1.30," http://www.nirsoft.net/utils/ese_database_view.html

[5] H. Chivers and C. Hargreaves, "Forensic data recovery from the windows search database," Digital Investigation, vol. 7, no. 3-4, pp. 114-126, Apr. 2011.

[6] H. Chivers, "Private browsing: A window of forensic opportunity," Digital Investigation, vol. 11, no. 1, pp. 20-29, Mar. 2014.

[7] JM. Gordon, "A forensic examination of windows desktop search (version 3)," Master's Thesis, Cranfield University, 2009

[8] J. Douglas, "Forensic artefacts present in microsoft windows desktop search," Master's Thesis, Cranfield University, 2009

〈저자소개〉

김 정 현 (Jeong-hyeon Kim) 학생회원
2008년 2월: 홍익대학교 컴퓨터공학과 졸업
2008년 3월~현재: 고려대학교 정보보호대학원 정보보호학과 석사과정
〈관심분야〉 디지털 포렌식, 모바일 보안

최 종 현 (Jong-Hyun Choi) 정회원
2012년 2월: 경희대학교 전자정보대학 컴퓨터공학과 공학사
2014년 8월: 고려대학교 정보보호대학원 정보보호학과 석사
2014년 9월~현재: 고려대학교 정보보호대학원 정보보호학과 박사과정
〈관심분야〉 디지털 포렌식

이 상 진 (Sang-jin Lee) 종신회원
1987년 2월: 고려대학교 수학과 학사
1989년 2월: 고려대학교 수학과 석사
1994년 8월: 고려대학교 수학과 박사
1989년 10월~1999년 2월: ETRI 선임 연구원
1999년 3월~2001년 8월: 고려대학교 자연과학대학 조교수
2001년 9월~현재: 고려대학교 정보보호대학원 교수
2008년 3월~현재: 고려대학교 디지털포렌식연구센터 센터장
〈관심분야〉 디지털 포렌식, 심층 암호, 해쉬 함수