情報保護學會論文誌
第20卷 第6號, 2010. 12

비할당 영역 데이터 파편의 문서 텍스트 추출 방안에 관한 연구*

유 병 영,† 박 정 흠, 방 제 완, 이 상 진‡
고려대학교 디지털 포렌식 연구센터

A Study on Extracting the Document Text for Unallocated Areas of Data Fragments*

Byeongyeong Yoo,† Jungheum Park, Jewan Bang and Sangjin Lee‡
Digital Forensics Research Center, Korea University

요 약

디지털 포렌식 관점에서 디스크의 비할당 영역의 데이터를 분석하는 것은 삭제된 데이터를 조사할 수 있다는 점에서 의미가 있다. 파일 카빙(Carving)을 이용하여 비할당 영역의 데이터를 복구할 경우 일반적으로 연속적으로 할당된 완전한 파일은 복구 가능하지만, 비연속적으로 할당되거나 완전하지 않은 형태의 단편화된 데이터 파편(Fragment)은 복구하기 어렵다. 하지만 데이터 파편은 많은 양의 정보를 포함하고 있기 때문에 이에 대한 분석이 필요하다. Microsoft Word, Excel, PowerPoint, PDF 문서 파일은 텍스트와 같은 정보들을 압축된 형태로 저장하거나 문서 내부에 특정 형식을 이용하여 저장한다. 앞서 언급한 문서 파일의 일부분이 데이터 파편에 저장되어 있을 경우 해당 데이터 파편에서 데이터의 압축 여부를 판단하거나 문서 내부 형식을 이용하여 텍스트 추출이 가능하다. 본 논문에서는 비할당 영역 데이터 파편에서 특정 문서파일의 텍스트를 추출하는 방안을 제시한다.

ABSTRACT

It is meaningful to investigate data in unallocated space because we can investigate the deleted data. Consecutively complete file recovery using the File Carving is possible in unallocated area, but noncontiguous or incomplete data recovery is impossible. Typically, the analysis of the data fragments are needed because they should contain large amounts of information. Microsoft Word, Excel, PowerPoint and PDF document file's text are stored using compression or specific document format. If the part of aforementioned document file was stored in unallocated data fragment, text extraction is possible using specific document format. In this paper, we suggest the method of extracting a particular document file text in unallocated data fragment.

Keywords: Digital Forensics, File Carving, Data Fragment, Text Extraction

접수일(2010년 6월 29일), 게재확정일(2010년 10월 3일)
* 본 연구는 지식경제부 및 한국산업기술평가 관리원의 산업원천기술개발사업의 일환으로 수행되었음 [10035157, 실시간 분석을 위한 디지털 포렌식 기술 개발]

† 주저자, pinpanel@korea.ac.kr
‡ 교신저자, sangjin@korea.ac.kr

I. 서 론

디지털 기기의 확산으로 대부분의 정보가 디지털 형태로 저장되고 있다. 이에 따라 각종 범죄 수사에 디지털 증거의 중요성이 커지고 있다. 디지털 데이터는 위변조가 쉽기 때문에 전통적인 관리 방식으로 쉽게 훼손될 가능성이 크므로 법적 증거로 사용되기 위해서는 세심한 관리가 필요하다. 따라서 디지털 증거의 수집, 가공, 분석, 처리를 위한 기술적, 절차적 문제를 다루기 위해 디지털 포렌식이 대두되었다. 디지털 포렌식은 범죄와 관련된 디지털 기기의 저장 데이터를 분석하여 실체적 진실을 밝히고 법정에서 유효한 증거로 채택되도록 하는데 그 목적이 있다.

디지털 포렌식 수사 시 저장매체의 데이터를 분석하는 것은 중요한 의미를 갖는다. 저장매체의 분석은 할당 영역 조사와 비할당 영역조사로 나눌 수 있다. 할당 영역 분석은 데이터가 완전한 형태로 존재하기 때문에 기존 응용프로그램을 이용하여 쉽게 분석이 가능하다. 이에 반해 비할당 영역 분석은 데이터가 완전한 형태로 존재하지 않는 경우가 많고, 메타데이터를 이용하여 데이터의 형식을 구분할 수 없기 때문에 분석에 어려움이 따른다.

일반적으로 비할당 영역을 분석하기 위해서는 파일 카빙기술을 이용하여 데이터를 복구한 후, 복구한 데이터를 분석한다. 파일 카빙은 파일시스템의 정보 없이 비할당 영역에서 파일을 추출하는 기법으로 데이터 복구나 디지털 포렌식 분야에서 이용된다. 카빙은 저장매체의 공간 할당에 따라 연속적인 카빙과 비연속적인 카빙으로 나눌 수 있다 [1].

연속적인 카빙은 파일의 전체 데이터가 저장매체에 연속적으로 기록된 경우에 사용하는 카빙 기법으로 파일의 고유한 시그니처나 파일구조를 기반으로 수행된다. 반면, 비연속적인 카빙 기법은 파일의 전체 데이터가 저장매체에 비연속적으로 조각나서 기록된 경우에 사용 되는 기법이다. 일반적으로 비할당 영역의 데이터가 연속적으로 완전하게 할당된 파일은 복구가 가능하지만, 단편화 되어 비연속적으로 할당되거나 다른 데이터에 의해 파일의 일부가 덮여 쓰인 경우에는 복구가 어렵다. 즉 비할당 영역에 존재하는 데이터 파편의 조합으로 하나의 완전한 파일을 복구해 낼 수 있는 경우를 제외하고는, 비할당 영역에 존재하는 데이터 파편들의 분석은 매우 어렵다. 비할당 영역의 데이터 파편은 디지털 포렌식 수사 시 분석의 어려움으로 인하여 생략되는 경우가 많으며, 중요한 내용을 포함하는 경우가 많기 때문에 파일 파편에 대한 조사 및 분석 기술을 제시하는 것은 디지털 포렌식 측면에서 큰 의미를 갖는다.

본 논문에서는 파일 카빙이 아닌, 파편 자체에 대한 분석으로 의미 있는 정보를 추출하는 것을 목적으로 한다. 데이터 파편에서 일반적인 유니코드, 아스키 인코딩 형태로 저장된 텍스트는 추출이 가능하지만, 특정 문서 포맷으로 저장된 텍스트는 추출이 어렵다. 하지만, Microsoft Word 2007(이하 Word 2007), Microsoft Excel 2007 (이하 Excel 2007), Microsoft PowerPoint 2007(이하 PowerPoint 2007), Adobe PDF(이하 PDF) 문서 파일은 텍스트와 같은 정보들을 압축된 형태로 저장하거나 문서 내부에 특정 형식을 이용하여 저장하는데, 이러한 특성을 이용하여 완전하지 않은 데이터 파편에서 텍스트 추출이 가능하다. 본 논문에서는 기존에 제시된 데이터 파편 분석에 대한 연구 및 디지털 포렌식과 관련된 사항을 서술하고, 비할당 영역 데이터 파편에서 Word 2007, Excel 2007, PowerPoint 2007, PDF 문서 파일의 텍스트를 추출하는 방안을 제시한다.

II. 관련 연구

데이터 파편 분석에 관하여 처음으로 K. Shanmugasundaram et al.이 연구하였다 [2]. 이 연구에서는 비할당 영역에서 데이터 파편을 수집한 후, 획득한 데이터 파편의 통계적인 특성을 이용하여 같은 파일의 일부라고 추정되는 파일 파편을 그룹화 한다. 그 후 각 그룹 내의 데이터 파편을 순서대로 재조합하여 완성된 하나의 파일로 복원한다. K. Shanmugasundaram et al.의 연구는 데이터 파편에서 하나의 완전한 파일을 분류해 낼 수 있는 방법을 제시한 것과 데이터 파편을 순서대로 정렬할 수 있는 방법론을 제시했다는 점에서 의미가 있지만, 데이터를 구성하는 일부 파편이 다른 데이터에 의해 덮여 쓰인 경우에는 완전하게 복구하기가 불가능하다.

따라서 원본 파일로의 복구보다는 현재 존재하는 데이터 파편에서 디지털 포렌식 관점의 의미 있는 데이터를 추출하는 과정이 필요하다. M. McDaniel et al.은 각 데이터 파편의 바이트별 빈도 특징과 같은 통계적 특성을 이용하거나, 데이터 파편 내에 존재하는 헤더/푸터 정보를 이용하여 데이터 파편의 원본 데이터가 어떠한 데이터인지를 판별한다 [3]. 데이터 파편의 원본 데이터를 판별하지만 파편내의 실제 데이터

추출방안은 고려되지 않았다.

데이터 파편에서 압축되어 있는 데이터 파편을 구별하고, 각각에 맞는 처리를 통하여 압축되지 않은 평문을 획득하는 방법이 이전에 연구되었다 [4]. 이 연구를 통해 압축된 데이터가 완전하지 않은 파편으로 존재하여도 이를 각 압축데이터의 특성을 이용하여 구분하고 압축을 해제하는 방법이 제시되었다. 압축 해제한 데이터에 평문 텍스트가 존재할 경우 텍스트 획득이 가능하다.

본 논문에서는 데이터 파편에서 압축된 데이터를 획득하는 방법을 이용하여 Word 2007, Excel 2007, PowerPoint 2007 파일의 텍스트를 추출하는 방안을 제시한다. 그리고 데이터 파편 내에 존재하는 PDF 본문영역의 시그니쳐 정보를 이용하여 PDF 파일의 텍스트를 추출하는 방안을 제시한다.

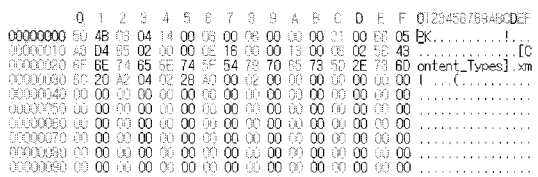
III. Microsoft Office 2007, Adobe PDF 파일의 텍스트 추출 방안

3.1 Microsoft Office 2007 텍스트 추출 방안

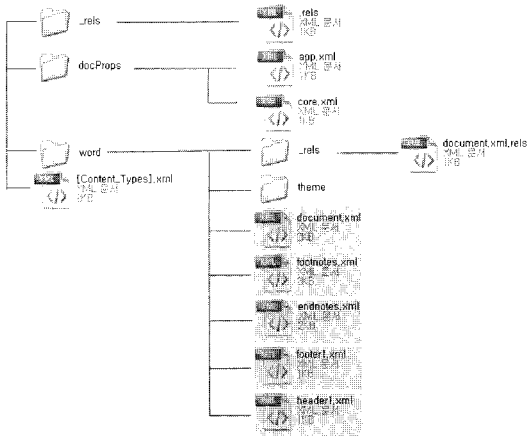
Microsoft Office 2007(이하 Office 2007) 파일은 [그림 1]과 같이 PK Zip 파일 형태로 압축되어 저장된다 [5]. PK Zip 파일의 시그니처인 0x504B0304를 헥사에디터를 이용하여 확인 할 수 있다.

[그림 2]는 Word 2007 파일의 압축을 해제한 후의 내부 구조이다. 데이터를 저장하기위해 xml 형식을 사용하며, 여러 폴더에 데이터를 나눠 저장한다. Ecxel 2007, PowerPoint 2007 파일은 Word 2007과 폴더 구조만 다르고 xml 형식을 사용하여 데이터를 저장하는 것은 동일하다.

Office 2007 파일이 단편화된 데이터 파편으로 저장되어 있을 경우 앞서 언급한 것처럼 PK Zip 파일 형식으로 저장되기 때문에 Office 2007 파일임을 구분하기 어렵다. 그러므로 Office 2007 데이터 파편에서 텍스트를 추출하기 위해서는 모든 데이터 파편에서 PK Zip 형태로 압축된 데이터를 수집한 후 압축을



[그림 1] Office 2007 파일 저장 형태



[그림 2] Word 2007 내부 파일 구조

해제한 데이터를 이용하여 Office 2007 데이터 파편임을 확인해야 한다. Office 2007 파일은 문서의 본문이나 속성정보와 같은 의미 있는 정보를 텍스트의 시작과 끝을 알리는 특정 구분자 사이에 저장하는데, 이를 이용하여 Office 2007 데이터 파편임을 확인할 수 있고 텍스트를 추출할 수 있다. 다음은 Word 2007, Excel 2007, PowerPoint 2007의 텍스트 저장 방법을 설명이다.

3.1.1 Microsoft Word 2007 텍스트 저장 방법

Word 2007 파일은 압축해제 후 생성된 폴더 중 word 폴더에 본문 텍스트를 저장하고 있는 파일이 위치하고 있다. word 폴더내의 파일 중 document.xml, endnotes.xml, footnotes. xml, header[x].xml, footer[x].xml 파일에 UTF-8 인코딩 방법으로 본문 텍스트가 저장된다 [6]. 각 파일의 저장되는 텍스트 종류는 [표 1]과 같다.

header[x].xml 파일은 머리글의 개수에 따라 파일이 여러 개 생성된다. 예를 들어 머리글의 개수가 세 개이면 header1.xml, header2.xml, header3.xml 파일이 생성된다. footer[x].xml 파일은 머리글과 마찬

[표 1] 파일별 저장 텍스트 종류

파일 이름	텍스트 종류
document.xml	본문, 목차 메모, 텍스트 상자 등
endnotes.xml	미주
footnotes.xml	각주
header[x].xml	머리글
footer[x].xml	바닥글

```
- <w:r w:rsidR="007C351F">
  - <w:rPr>
      <w:rFonts w:ascii="돋움" w:eastAsia="돋움" w:hAnsi="돋움" w:hint="eastAsia" />
      <w:sz w:val="22" />
    </w:rPr>
    <w:t>여러 가지</w:t>
  </w:r>
```

〔그림 3〕 Word 2007 본문 텍스트 저장 방법 1

```
- <w:r w:rsidRPr="00E652EE">
  - <w:rPr>
      <w:rFonts w:ascii="돋움" w:eastAsia="돋움" w:hAnsi="돋움" w:hint="eastAsia" />
      <w:sz w:val="22" />
    </w:rPr>
    <w:t xml:space="preserve">Product ID :</w:t>
  </w:r>
```

〔그림 4〕 Word 2007 본문 텍스트 저장 방법 2

```
- <w:p w:rsidR="009A6796" w:rsidRDefault="00DF728F">
  - <w:r>
    - <w:rPr>
        <w:rFonts w:hint="eastAsia" />
      </w:rPr>
      <w:t><></w:t>
    </w:r>
  </w:p>
```

```
0000200 77 3A 72 3E 3C 77 3A 72 50 72 3E 3C 77 3A 72 46 |w:r><w:rPr><w:rF
0000210 6F 6E 74 73 20 77 3A 68 69 6E 74 3D 22 65 61 73 |onts w:hint="eas
0000220 74 41 73 69 61 22 2F 3E 3C 2F 77 3A 72 50 72 3E |tAsia"/></w:rPr>
0000230 3C 77 3A 74 3E 26 6C 74 3B 26 67 74 3B 3C 2F 77 |<w:t>&lt;&gt;</w
0000240 3A 74 3E 3C 2F 77 3A 72 3E 3C 2F 77 3A 70 3E 3C |:t></w:r></w:p><
0000250 77 3A 73 65 63 74 50 72 20 77 3A 72 73 69 64 52 |w:sectPr w:rsidR
0000260 3D 22 30 30 39 41 36 37 39 36 22 20 77 3A 72 73 |="009A6796" w:rs
```

〔그림 5〕 '<', '>' 저장 방법

```
<dc:title>제목 입력</dc:title>
<dc:subject>주제 입력</dc:subject>
<dc:creator>포켄석</dc:creator>
<cp:keywords>키워드 입력</cp:keywords>
<dc:description>설명입력 ... _x005f_x005f_</dc:description>
<cp:lastModifiedBy>빌 게이츠</cp:lastModifiedBy>
<cp:revision>1</cp:revision>
<dcterms:created xsi:type="dcterms:W3CDTF">2010-04-21T02:03:00Z</dcterms:created>
<dcterms:modified xsi:type="dcterms:W3CDTF">2010-04-21T02:03:00Z</dcterms:modified>
<cp:category>범주 입력</cp:category>
</cp:coreProperties>
```

〔그림 6〕 Word 2007 문서 속성 저장 정보

가지로 바닥글의 개수에 따라 파일이 여러 개 생성된다.

xml 파일에 저장되는 본문 텍스트는 텍스트의 시작과 종료를 알리는 특정 형식을 이용하여 구분이 가능하다. "〈w:t〉"와 "〈w:t xml:space= "preserve"〉"는 텍스트의 시작을 나타내고, "〈/w:t〉"는 텍스트의 종료를 나타낸다. 〔그림 3〕과 〔그림 4〕는 두 가지 방식의 텍스트 저장 방법을 나타낸다.

만약 텍스트에 '<', '>' 문자가 저장되어도 xml 저장

```
0000190 3C 2F 63 70 3A 6B 65 79 77 6F 72 64 73 3E 3C 64 |</cp:keywords><d
00001A0 63 3A 64 65 73 63 72 69 70 74 69 6F 6E 3E EC 84 |c:description>..
00001B0 A4 EB AA 85 EC 9E 85 EB A0 A5 20 20 20 20 20 20 |.........
00001C0 20 20 20 20 20 20 20 20 20 20 72 6F 5F 78 30 30 |          pro_x00
00001D0 30 64 5F 0D 0A 70 72 6F 20 20 20 20 20 20 20 20 |0d_..pro
00001E0 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 |
00001F0 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 |
0000200 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 2E |
0000210 2E 20 20 20 20 20 5F 78 30 30 35 66 5F 78 30 30 |.     _x005f_x00
0000220 35 66 5F 3C 2F 64 63 3A 64 65 73 63 72 69 70 74 |5f_</dc:descript
0000230 69 6F 6E 3E 3C 70 3A 6C 61 73 74 4D 6F 64 69 |ion><cp:lastModi
```

〔그림 7〕 줄 바꿈 구분자 저장 방법

방식에 따라 '<', '>'로 바꾸어 저장하기 때문에 텍스트의 시작과 종료를 구분하는데 문제가 발생하지 않는다. 〔그림 5〕는 '<', '>' 문자의 저장 방식을 나타낸다.

문서속성 정보는 docProps 폴더의 core.xml 파일에서 문서속성 정보가 저장되는 것을 확인할 수 있다. 〔그림 6〕은 core.xml에 저장된 문서 속성 정보이다. 문서 속성에는 제목, 주제, 만든 이, 범주, 키워드, 설명, 생성 시간, 수정 시간, 마지막으로 수정한 사용자 등의 정보가 저장된다. 문서속성 텍스트의 획득은 본문 텍스트와 마찬가지로 텍스트의 시작과 종료를 알리는 특정 형식을 이용하여 획득이 가능하다. 〔표 2〕는 각 문서속성별 텍스트의 시작, 종료 구분 형식이다.

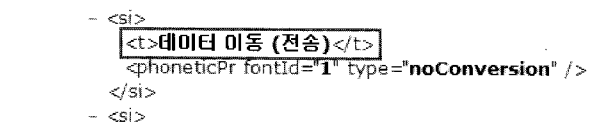
문서 속성 중 설명은 텍스트에 줄 바꿈을 구분하기 위해 "_x000d_" 문자열을 저장한다. 이런 경우, 줄 바꿈 구분 문자열 "_x000d_"과 텍스트 "_x000d_" 문자열을 구분하기 위하여, 텍스트 "_x000d_" 문자열을 저장할 때는 "_x005f_x000d_" 형태로 저장되며, 텍스트 "_x005f_" 문자열은 저장할 때는 "_x005f_x005f_" 형태로 저장된다. 〔그림 7〕은 앞서 언급한 줄 바꿈에 대한 저장 방법을 보여준다.

3.1.2 Microsoft Excel 2007 텍스트 저장 방법

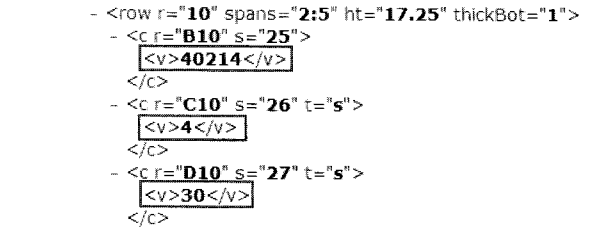
Excel 2007 파일은 압축 해제 후 생성된 폴더 중 xl 폴더에 본문 텍스트를 저장하고 있는 파일이 위치

〔표 2〕 문서 속성별 텍스트 구분 형식

문서속성	텍스트 시작	텍스트 종료
제목	〈dc:title〉	〈/dc:title〉
주제	〈dc:subject〉	〈/dc:subject〉
만든 이	〈dc:creator〉	〈/dc:creator〉
키워드	〈cp:keywords〉	〈/cp:keywords〉
설명	〈dc:description〉	〈/dc:description〉
범주	〈dc:category〉	〈/dc:category〉
마지막 수정자	〈cp:lastModifiedBy〉	〈/cp:lastModifiedBy〉
생성 시간	〈dcterms:created xsi:type="dcterms:W3CDTF"〉	〈/dcterms:created〉
수정 시간	〈dcterms:modified xsi:type="dcterms:W3CDTF"〉	〈/dcterms:modified〉



〔그림 8〕 sharedString.xml 파일 텍스트 저장 방법



〔그림 9〕 sheet〔x〕.xml 파일 텍스트 저장 방법

하고 있다. xl 폴더내의 파일 중 sharedString.xml 파일과 xl\worksheets 폴더의 sheet〔x〕.xml 파일에 UTF-8 인코딩 방법으로 본문 텍스트가 저장된다〔6〕. sheet〔x〕.xml 파일은 해당 Excel 문서가 포함하고 있는 Worksheet의 개수만큼 생성된다. sharedString.xml 파일에는 셀 표시 형식이 지정되지 않은 모든 문자열이 저장된다. Word 2007과 마찬가지고 xml 파일에 저장되는 본문 텍스트는 텍스트의 시작과 종료를 알리는 특정 형식을 이용하여 구분이 가능하다. sharedString.xml 파일에서 텍스트의 시작은 "〈t〉"로 구분하고 텍스트의 종료는 "〈/t〉"로 구분한다. 〔그림 8〕은 sharedString.xml 파일의 텍스트 저장 방법이다.

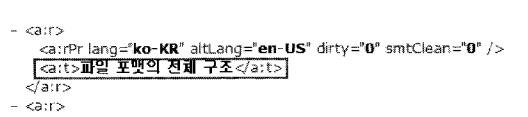
sheet〔x〕.xml 파일에는 셀 표시 형식이 지정된 숫자 형식의 텍스트가 저장된다. 텍스트의 시작은 "〈v〉"로 구분하고 텍스트의 종료는 "〈/v〉"로 구분한다. 〔그림 9〕는 sheet〔x〕.xml 파일의 텍스트 저장 방법을 나타낸다. 문서 속성의 저장 방법은 Word 2007과 동일하게 저장 된다.

3.1.3 Microsoft PowerPoint 2007 텍스트 저장 방법

PowerPoint 2007 파일은 압축 해제 후 생성된 폴더 중 ppt 폴더에 본문 텍스트를 저장하고 있는 파일이 위치하고 있다. ppt 폴더내의 파일 중 slide〔x〕.xml, notesMaster〔x〕.xml, notes Slide〔x〕.xml, slideMaster〔x〕.xml, slide Layout〔x〕.xml, handoutMaster〔x〕.xml, comment〔x〕.xml, data〔x〕.xml 파일에 UTF-8 인코딩 방법으로 본문 텍스트가 저장된다〔6〕. 각 파일에 저장되는 텍스트의 종류는 〔표 3〕과 같다.

〔표 3〕 PowerPoint 2007 파일별 저장 텍스트 종류

파일 이름	텍스트 종류
slide〔x〕.xml	슬라이드 본문
notesMaster〔x〕.xml	슬라이드 노트 마스터
notesSlide〔x〕.xml	슬라이드 노트
slideLayout〔x〕.xml	슬라이드 마스터의 레이아웃
handoutMaster〔x〕.xml	유인물
comment〔x〕.xml	메모상자
data〔x〕.xml	다이어그램



〔그림 10〕 PowerPoint 2007 본문 텍스트 저장 방법

〔그림 11〕 PowerPoint 2007 메모 텍스트 저장 방법

slide〔x〕.xml, notesSlide〔x〕.xml 파일은 슬라이드의 개수에 따라 파일이 여러 개 생성된다. 예를 들어 슬라이드의 개수가 세 개이면 slide1.xml, slide2.xml, slide3.xml 파일이 생성된다. notesMatser〔x〕.xml, slideLayout〔x〕.xml, handoutMaster〔x〕.xml, comment〔x〕.xml, data〔x〕.xml 파일은 해당 개체의 개수에 따라 파일이 여러 개 생성된다.

xml 파일에 저장되는 본문 텍스트는 Word 2007과 마찬가지로 텍스트의 시작과 종료를 알리는 특정 형식을 이용하여 구분이 가능하다. comment〔x〕.xml 파일을 제외한 다른 파일의 텍스트 시작 구분자는 "〈a:t〉"이고 텍스트 종료 구분자는 "〈/a:t〉"이다. comment〔x〕.xml 파일의 텍스트 시작 구분자는 "〈a:text〉"이고 텍스트 종료 구분자는 "〈/a:text〉" 이다. 〔그림 10〕은 comment〔x〕.xml 파일을 제외한 본문 텍스트 저장 방법이고 〔그림 11〕은 comment〔x〕.xml 파일의 메모 텍스트 저장 방법이다. 문서 속성의 저장 방법은 Word 2007 과 동일하다.

3.2 Adobe PDF 텍스트 추출 방안

PDF 파일은 본문 텍스트를 deflate 압축 알고리즘을 이용해서 압축하여 저장한다. 텍스트 압축 블록의 시작부분에 "stream" 이라는 아스키 시그니처를 가지며, 텍스트 압축 블록의 끝부분에는 "endstream"이라는

〔그림 12〕 PDF 파일 텍스트 블록

```
BT
   /F13 48 Tf 20 40 Td
   0 Tr 0.5 g (ABC) Tj
ET
```

〔그림 13〕 PDF 파일 텍스트저장 형식

아스키 시그니처를 갖는다〔7〕. 〔그림 12〕는 deflate로 압축된 텍스트 블록과 텍스트 압축 블록의 시그니처를 나타낸다.

텍스트 압축 블록의 압축을 해제하면 본문 텍스트 정보를 획득할 수 있다. 텍스트 저장 정보는 실제 텍스트의 내용과 글자 크기, 폰트 등의 텍스트 속성 정보를 포함한다. PDF 파일은 텍스트를 저장할 때 Adobe PS ISOLatin1 인코딩과 유니코드 인코딩 두 가지 방법을 사용한다. Adobe PS ISOLatin1 인코딩은 영어 문자와 라틴 문자의 표현이 가능하다. 따라서 영어와 라틴문자로 본문을 구성할 경우에는 Adobe PS ISOLatin1 Encoding이 사용되는데 텍스트 압축 블록의 압축을 해제한 데이터에 '( )' 안에 저장되는 데이터가 실제 Adobe PS ISOLatin1 인코딩을 사용한 텍스트 이다〔7〕. 〔그림 13〕은 Adobe PS ISOLatin1 인코딩을 사용하여 저장된 텍스트를 나타낸다.

유니코드 인코딩은 앞서 언급한 Adobe PS ISOLatin1 인코딩만으로 표현될 수 없는 문자가 존재할 경우 사용된다. PDF에서 사용되는 유니코드 인코딩은 유니코드 각 문자에 대응되는 별도의 맵핑 테이블을 사용하여 문자를 인코딩 한다. 따라서 일반적으로 사용되는 유니코드 인코딩과는 데이터가 다르다. 유니코드는 텍스트 압축 블록의 압축을 해제한 데이터에 '< >' 안에 저장되어 있다〔7〕.

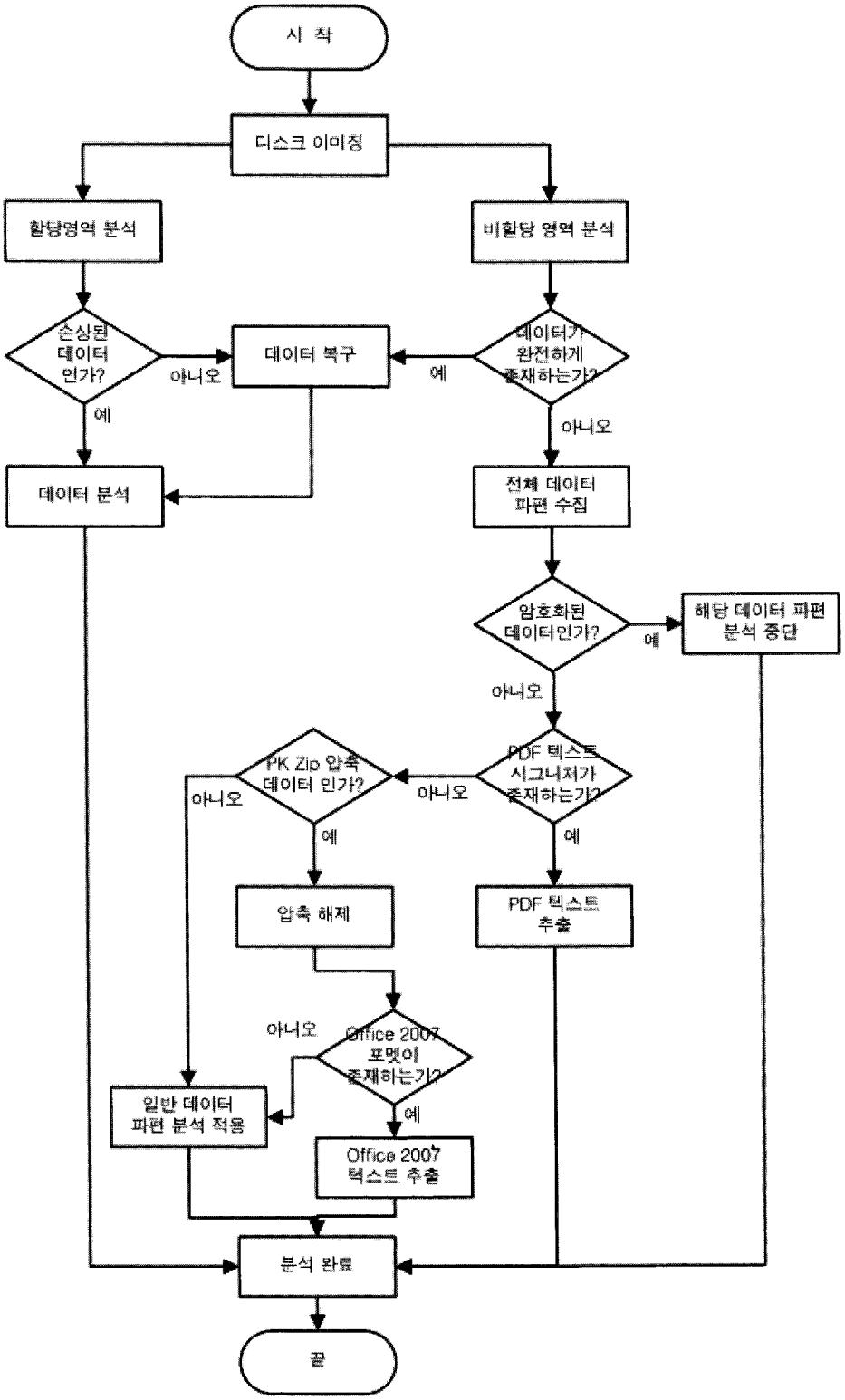
데이터 파편에 PDF 파일의 일부분이 존재할 경우 텍스트 압축 블록의 시작 시그니처 "stream"과 종료 시그니처 "endstream"을 이용하여 텍스트 압축 블록의 획득이 가능하다. 획득한 텍스트 압축 블록을 deflate 압축 알고리즘으로 압축 해제한 후에 데이터를 확인하여 텍스트 추출할 수 있다. Adobe PS ISOLatin1 인코딩을 사용한 경우에는 압축 해제한 데이터에 '( )' 안에 존재하는 텍스트만을 쉽게 추출할 수 있다. 하지만 유니코드 인코딩을 사용한 경우에는 텍스트를 디코딩하기 위해 유니코드 맵핑 테이블이 필요하다. 일반적으로 맵핑 테이블은 고정된 위치에 존재하지 않기 때문에, 데이터 파편에서 해당 유니코드에 대응되는 맵핑 테이블을 찾을 가능성은 매우 낮다. 따라서 PDF파일의 데이터 파편에서 유니코드 텍스트 추출은 어려움이 따른다.

IV. Microsoft Office2007, Adobe PDF 데이터 파편의 텍스트 추출 절차

디지털 포렌식 수사 시 압수한 디스크 이미지를 조사 및 분석하기 위해서는 디스크의 할당 영역과 비할당 영역을 조사해야 한다. 할당 영역에 정상적으로 존재하는 데이터의 조사는 EnCase와 같은 도구를 이용하여 실제 파일의 내용 및 메타데이터를 조사하고, 손상된 데이터는 데이터 복구 등을 이용하여 정상적인 파일로 복구하여 조사한다. 비할당 영역에 데이터를 조사하기 위해서는 일반적으로 파일 카빙이나 데이터 복구 도구를 이용하여 데이터가 손상되지 않은 완전한 파일을 복구하여 조사한다. 이러한 데이터를 제외한 나머지 데이터 파편들을 조사하는 방법은 데이터 상에 존재하는 일반 문자열을 추출하는 방법이 있다. 그리고 앞서 언급한 것처럼 Office 2007과 PDF 파일은 데이터 파편에 존재하는 데이터만으로도 문서의 고유한 포맷을 이용하여 본문 텍스트 추출이 가능하다. 〔그림 14〕는 본 논문에서 제시하는 데이터 파편을 분석하는 알고리즘이다.

비할당 영역에서 획득한 데이터 파편에 대하여, 일반적인 데이터일 경우 텍스트를 추출하고 통계 분석이나 시그니처 분석을 행하는 것은 지금까지 공개되어 온 방법이다. 그리고 완전하지 않은 압축 데이터를 판별하고 압축을 해제하여 분석을 수행하는 방법도 공개되었다. 〔그림 14〕의 알고리즘에서는 데이터 파편 분석에 대하여 지금까지 공개된 방법 이외에 3장에서 언급한 Office 2007과 PDF 파일의 텍스트 저장 구조를 이용하여 텍스트를 추출하는 방안을 적용하였다.

먼저 디스크 할당영역과 비할당 영역에서 데이터가 완전하게 존재하는 부분을 제외한 모든 데이터 파편을 수집한다. 수집한 데이터 파편이 암호화 되었을 경우에는 의미 있는 정보 추출이 거의 불가능하기 때문에 분석 대상에서 제외한다. 암호화된 데이터 파편을 제외한 나머지 데이터 파편에서 PDF 파일의 텍스트 저



〔그림 14〕 데이터 파편 분석 알고리즘

장 블록의 시그니처가 존재하는지 탐지한다. 만약 시그니처가 존재하고 해당 데이터가 deflate 알고리즘으로 압축되어있을 경우 3장에서 언급한 방법으로 PDF 텍스트를 추출한다. 이와 동시에 데이터 파편의 PK Zip 압축 여부를 파악하여 Office 2007 데이터 파편의 텍스트를 추출한다. PK Zip 데이터 파편이 발견될 경우 이를 압축해제 하여 3장에서 언급한 Office 2007의 텍스트 저장 방법을 이용하여 해당 데이터에서 텍스트를 추출할 수 있다. 압축 해제한 데이터에 Office 2007의 텍스트가 발견되지 않을 경우에는 일반 텍스트를 추출한다. 그리고 PK Zip으로 압축되어있지 않고 PDF 파일의 텍스트 저장 블록 시그니처가 존재 하지 않을 경우에도 일반 텍스트를 추출한다.

V. 결 론

본 논문에서는 비할당 영역에 존재하는 데이터 파편을 분석함에 있어서 Office 2007과 PDF 파일의 텍스트 저장 구조를 이용하여 텍스트를 추출하는 방안을 제시하였다. Office 2007의 경우 PK Zip으로 데이터를 압축하여 저장하는데, 완전하지 않은 압축데이터를 압축 해제하여 텍스트 시작과 끝의 구분자를 이용하여 텍스트 추출이 가능하다. PDF 파일은 텍스트를 deflate 알고리즘으로 압축한 일정 크기의 블록에 저장하는데, 압축 블록의 시작과 끝을 구분할 수 있는 시그니처가 존재하였다. 이를 이용하여 압축을 해제하여 텍스트 저장 구조를 확인하여 텍스트 추출이 가능하다. Office 2007, PDF 파일은 본문 텍스트를 저장할 때 텍스트의 시작과 끝에 구분자가 존재하기 때문에 파일 전체의 데이터가 아닌 데이터 파편만으로도 텍스트 추출이 가능하다.

지금까지 디지털 포렌식 수사 시 비할당 영역에서 데이터 파편을 분석하는 경우는 완전한 파일로 복구가 가능한 데이터 파편에 대한 방법론의 연구가 주로 이루어졌다. 반면 완전한 파일로 복구가 불가능한 데이터 파편에 대해서는 유니코드나 아스키코드 등의 일반적인 문자열 인코딩을 사용한 텍스트만을 추출하였다. 하지만 데이터 파편에서 특정 파일의 구조를 이용하여 데이터를 추출하는 방법을 본 논문에서 제시하였다. 제시한 데이터 파편 분석 알고리즘을 이용하여 실제 하드 디스크에서 텍스트 추출을 실시한 결과 상당한 양의 텍스트를 획득할 수 있었다. 하드 디스크 용량이 증가함에 따라 비할당 영역에서 데이터 파편을 분석하는 경우 많은 시간이 소요된다. 따라서 본 논문에서 제시한 알고리즘을 이용하여 비할당 영역 데이터 파편 분석을 도구로 구현하여 자동화할 것이다. 그리고 한글과 컴퓨터 한글, Microsoft Office의 다른 버전들의 데이터 파편에서의 데이터 및 텍스트 추출 방안을 분석하여 디지털 포렌식 수사 시 디스크 비할당 영역의 데이터 파편 수사 모델을 제시할 것이다.

참고문헌

[1] 권태석, 변근덕, 이상진, 임종인 "포렌식 관점에서 효율적인 파일 카빙 알고리즘 설계 제안," 한국방송공학회, pp. 205-208, 2008년 2월.

[2] Kulesh Shnmugasundaram and Nasir Memon, "Automatic Reassembly of Decument Fragments via context Based Statistical Models," Proceedings of the 19th Annual Computer Security Applications Conference (ACSAC), pp. 152-159, 2003.

[3] Mason McDaniel and M. Hossain Heydari, "Content Basec File Type Detection Algorithms," 6th Annual Hawaii International Conference on System Sciences(HICSS), pp. 108-114, 2003.

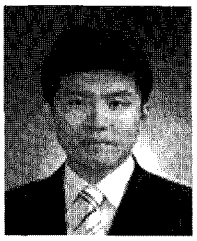
[4] 박보라, 이상진, "비할당 영역 데이터 파편의 압축 여부 판단과 압축 해제," 정보보호학회 논문지, 18(4), pp. 175-185, 2008년 8월.

[5] Frank Rice, Introducing the Office (2007) Open XML File Formats, Microsoft Corporation, URL: http://msdn2.microsoft.com/ko- kr/library/aa338205.aspx, 2006

[6] Microsoft Corporation, Office Open XML Part 4 - Markup Language Reference, Microsoft Corporation, 2006

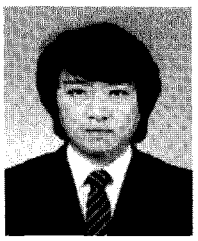
[7] Adobe Systems Incorporated, Document management - Portable document format - Part 1: PDF 1.7, Adobe Systems Incorporated, 2008.

〈著者紹介〉

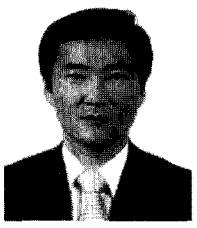


유 병 영 (Byeongyeong Yoo) 정회원
2009년 2월: 상명대학교 컴퓨터과학 학사
2009년 3월~현재: 고려대학교 정보경영공학전문대학원 석사과정
〈관심분야〉 디지털 포렌식, 파일 시스템

박 정 흠 (Jungheum Park) 정회원
2007년 2월: 한양대학교 정보통신대학 컴퓨터전공 공학사
2007년 3월~2009년 2월: 고려대학교 정보경영공학전문대학원 공학석사
2009년 3월~현재: 고려대학교 정보경영공학전문대학원 박사과정
〈관심분야〉 디지털 포렌식, 안티-안티 포렌식



방 제 완 (Jewan Bang) 정회원
2007년 2월: 한세대학교 정보통신공학 학사
2007년 3월~현재: 고려대학교 정보경영공학전문대학원 석박사통합과정
〈관심분야〉 디지털 포렌식, 소프트웨어 역공학 분석, 임베디드 시스템



이 상 진 (Sangjin Lee) 정회원
1987년 2월: 고려대학교 학사 졸업
1989년 2월: 고려대학교 석사 졸업
1994년 8월: 고려대학교 박사 졸업
1989년 10월~1999년 2월: ETRI 연구원 역임
1999년 10월~현재: 고려대학교 정교수
1997년 12월: 국가안전기획부장 표창
〈관심분야〉 디지털 포렌식, 모바일 포렌식, 심층 암호, 해쉬 함수