情報保護學會論文誌
第18卷 第4號, 2008. 8

# 비할당 영역 데이터 파편의 압축 여부 판단과 압축 해제*

박 보 라†, 이 상 진‡
고려대학교 정보보호기술연구센터

# Determinant Whether the Data Fragment in Unallocated Space is Compressed or Not and Decompressing of Compressed Data Fragment*

Bora Park†, Sangjin Lee‡
Center for Information Security Technologies, Korea University

## 요 약

컴퓨터 포렌식 관점에서 디스크의 비할당 영역(unallocated space)에 존재하는 데이터를 분석하는 것은 삭제된 데이터를 조사할 수 있다는 점에서 의미가 있다. 하지만 대부분의 경우에 비할당 영역에 존재하는 데이터는 응용 프로그램으로 읽을 수 있는 완전한 파일의 형태가 아닌 단편화된 파편(Fragment)으로 존재하며 이는 암호화되거나 압축된 형식으로 존재하기도 한다. 특히 데이터의 일부만 남아있고 나머지는 다른 데이터로 덮여 쓰인 상태의 데이터 파편을 분석하는 것은 매우 어려운 일이며, 특히 존재하는 데이터 파편이 압축되거나 암호화된 경우에는 데이터가 랜덤(Random)한 특성을 가지기 때문에 통계 분석이나 시그니처 분석과 같은 기존의 데이터 파편 분석 방법만으로는 의미 있는 정보를 획득할 수 없게 된다. 따라서 파일 파편의 압축 및 암호화 여부를 판단하는 선 처리 작업이 필요하며 압축된 파편은 압축 해제를 시도해야 한다. 압축 해제로서 획득한 평문 데이터 파편은 기존에 제시된 데이터 파편 분석 방식으로 분석할 수 있다. 본 논문에서는 컴퓨터 포렌식 수사 시 비할당 영역에 존재하는 파일 파편의 분석 기술에 대해 서술한다.

## ABSTRACT

It is meaningful to investigate data in unallocated space because we can investigate the deleted data. However the data in unallocated space is formed to fragmented and it cannot be read by application in most cases. Especially in case of being compressed or encrypted, the data is more difficult to be read. If the fragmented data is encrypted and damaged, it is almost impossible to be read. If the fragmented data is compressed and damaged, it is very difficult to be read but we can read and interpret it sometimes. Therefore if the computer forensic investigator wants to investigate data in unallocated space, formal work of determining the data is encrypted of compressed and decompressing the damaged compressed data. In this paper, I suggest the method of analyzing data in unallocated space from a viewpoint of computer forensics.

**Keywords** : data fragment, compressed data recovery

## Ⅰ. 서 론

현대에는 다양한 정보기기들이 활용되고 있고 이에 따른 정보의 생산 및 유통에 있어서 95% 이상이 디지털 형태이다. 따라서 범죄 수사에 있어 물리적 형태의 증거뿐만 아니라 전자적 증거(Electronic evidence)를 다루는 컴퓨터 포렌식(Computer Forensics) 분야가 점차 확대되고 있다[1]. 즉, 현대에는 '범죄 수사에 사용되는 과학적인 증거 수집 기법 및 증거 분석 기법'이라는 전통적 의미의 포렌식에서 나아가 '디지털 증거 수집 및 분석 기술을 적용하여 범행과 관련된 디지털 데이터와 기록 등을 증거로 수집 및 분석'하는 컴퓨터 포렌식의 개념이 포렌식 분야의 중요한 위치를 차지하고 있다. 컴퓨터 포렌식의 간단한 절차는 다음과 같다. 우선 EnCase와 같은 도구를 사용하여 디스크 내에 존재하는 데이터를 수집하고 검색한다. 이 때 할당 영역에 정상적으로 존재하는 데이터는 해당 응용 프로그램을 이용하여 파일을 열어 봄으로써 쉽게 조사할 수 있다. 또한 비할당 영역에 존재하는 데이터를 조사함으로써 삭제되거나 은닉된 파일들을 복구하고 탐색하여 증거 파일이 될 만한 데이터를 분류해낸다. 특히 손상된 파일을 복구할 때는 파일의 형식(Format)에 기반을 둔 복구 방법을 사용할 수도 있고 시판되고 있는 파일 복구 도구를 사용할 수도 있다. 하지만 비할당 영역에 존재하는 파일은 단편화되어 연속적으로 존재하지 않을 수 있으며 때때로 파일의 일부분이 다른 데이터로 덮여 쓰인 경우도 많다. 이러한 경우 비할당 영역에 존재하는 파일을 파일 복구 도구로 복구해내기는 매우 어렵다. 즉, 비할당 영역에 존재하는 파일 파편의 조합으로 하나의 완전한 파일을 복구해 낼 수 있는 경우를 제외하고는, 비할당 영역에 존재하는 파일 파편들을 분석하는 기술 및 방법론이 현재로서는 제시되지 않았으며 포렌식 수사관들도 이러한 파일 파편에 대한 조사 및 분석을 생략하는 경우가 있다. 따라서 파일 파편을 분석하는 데에 필요한 기술을 제시하는 것은 컴퓨터 포렌식의 영역을 확장한다는 점에서 매우 의미가 있다.

접수일 : 2008년 4월 28일; 수정일 : 2008년 8월 18일;
채택일 : 2008년 6월 18일
* 본 연구는 지식경제부 및 정보통신연구진흥원의 IT신성장 동력핵심기술개발사업의 일환으로 수행하였음. [2007-S019-02, 정보투명성 보장형 디지털 포렌식 시스템 개발]
† 주저자, danver123@korea.ac.kr
‡ 교신저자, sangjin@korea.ac.kr

비할당 영역에 존재하는 데이터 중에서 파일 파편을 조사하고 분석함으로써 얻고자 하는 것은 파일의 메타데이터나 텍스트 데이터이다. 파일 파편을 이용하여 하나의 완전한 파일을 생성해내기 위해서는 파일 카빙(File Carving) 기술이 필요하다. 본 논문에서는 파일 카빙이 아닌, 파편 자체에 대한 분석으로 의미 있는 정보를 추출하는 것을 목적으로 한다. 때때로 파일 파편은 압축되거나 암호화된 형태로 존재하며 이는 파일 파편에 대한 분석을 더욱 어렵게 한다. 암호화된 파일 파편은 의미 있는 정보 추출이 거의 불가능하기 때문에 분석 대상에서 제외하고, 압축된 파일 파편은 그 것이 손상된 압축 데이터라 하더라도 데이터에 사용된 압축 알고리즘의 특징을 이용하여 압축 해제를 시도해 볼 수 있다. 압축이 풀린 데이터는 파일 형식 등에 근거하여 분석이 용이하게 된다.

본 논문은 다음과 같이 구성되어 있다. 우선 기존에 제시된 파일 파편 분석에 대한 연구 및 컴퓨터 포렌식에 관련된 사항을 다음 장에서 서술한다. 또한 비할당 영역에서 클러스터 단위로 추출한 데이터에 대하여 압축 및 암호화 여부를 판단하는 방법, 파일 파편의 분류 기준 및 손상된 압축 데이터의 압축 해제와 압축이 풀린 데이터를 분석하여 텍스트를 추출할 수 있는 방법을 서술하고, 마지막으로 결론 및 향후 계획에 대하여 서술한다.

## Ⅱ. 관련 연구

파일 파편 역시 컴퓨터 포렌식 수사에서 중요한 디지털 증거 중의 하나이다[1]. 디지털 증거는 쉽게 파편화될 수 있고 컴퓨터 포렌식 수사관은 종종 이러한 파일 파편을 찾을 수 있다. 따라서 증거를 복구하는 과정에서 완전한 파일로 복구되지 않는 비할당 영역에 존재하는 파일 파편을 처리하는 것은 보다 철저한 수사를 위하여 필요한 기술이다.

파일 파편 분석에 대해 처음으로 *K. Shanmugasundaram et al.*이 연구하였으며[2], 이 연구에서 제시하고 있는 파일 파편에 대한 분석 순서는 다음과 같다. 우선 비할당 영역에 존재하는 파일 파편을 수집한다. 이 때, 암호화되거나 압축된 데이터는 적절한 암호 공격이나 변환을 이용하여 압축되지 않은 평문을 획득한다. 그 다음으로 획득한 파일 파편의 통계적인 특성을 이용하여 같은 파일의 일부라고 추정되는 파일 파편을 그룹화(Grouping)한다. 그 다음 각 그룹 내의 파일 파편을 순서대로 재조합함으

로써 하나의 파일을 복원해 낸다(Reassembling).

이 연구에서는 같은 파일에서 생겨난 파일 파편을 분류해 낼 수 있는 기법을 제시한 것과 파일 파편을 순서대로 정렬할 수 있는 방법론을 제시한다는 것에서 의미가 있다. 파일을 완전히 재현하기 위하여 필요한 모든 파일 파편이 존재하기만 한다면 *K. Shanmugasundaram*가 제시하는 방법으로 파일 파편을 수집하여 원본 파일을 정상적인 형태로 복원할 수 있다. 하지만 파일을 구성하는 일부 파일 파편이 다른 데이터에 의하여 덮여 쓰인 경우에는 원본 파일을 완전하게 복원하는 것이 불가능한 경우가 많으며 따라서 이러한 경우에는 원본 파일로의 복구보다는 존재하는 파일 파편에서 의미 있는 데이터를 추출할 수 있는 지의 여부를 판단하고, 의미 있는 데이터를 추출해내는 작업이 효과적이다.

이렇게 파일 파편 자체를 조사하고 분석하는 기술에 대하여 *M. McDaniel et al.*이 연구하였다[3]. 즉, 각 파일 파편이 가진 바이트(byte)별 빈도 특징과 같은 통계적 특성을 이용하여 수집한 파일 파편의 원본 데이터가 어떠한 데이터인지를 판별하거나, 파일 파편 내에 존재하는 파일의 헤더/푸터(header/footer) 정보를 이용하여 원본 데이터에 따라 파일 파편을 분류하는 방법을 제시한다. 즉, 파일 파편의 'Fingerprint' 개념을 도입하면서 파일 파편을 분류하고 있다. 이러한 분류 방법에 따라 분류된 파일 파편은 원본 데이터의 포맷이나 특성에 따라 분석될 수 있다.

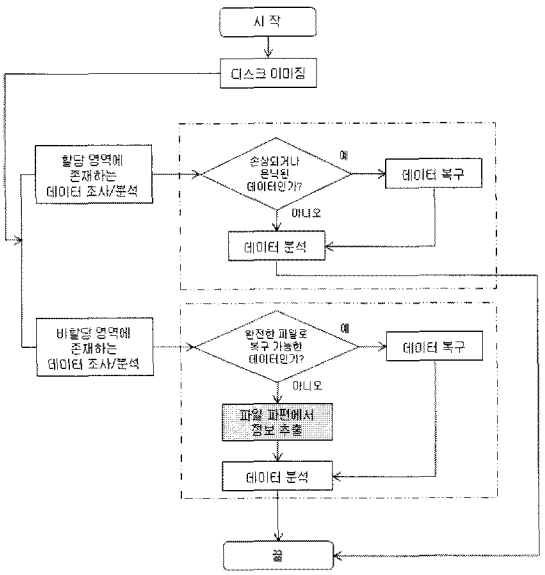
하지만 파일 파편이 압축되거나 암호화 된 경우의 분석 과정에 대해서는 더욱 연구가 필요하며 그러한 연구를 본 논문에서 진행한다. 즉, 파일 파편이 암호화되어 있는지, 압축되어 있는지 암호화와 압축 모두 되어 있지 않는 지를 명확하게 구별하는 작업과 랜덤하지 않은 데이터를 획득하는 작업이 *M. McDaniel*의 연구의 결과를 더욱 효용성 있게 한다. 이는 암호화나 압축은 원본 데이터의 엔트로피(Entropy)를 높임으로써 원본 데이터의 통계적 특성이나 원본 데이터가 가지고 있는 시그니처(signature)를 무력화시키기 때문이다.

최근 *B. Li et al.*[4]은 ESK(Enhanced String Kernel), ESA(Extended suffix array)라는 개념을 제시하면서 새로운 파일 파편 분석 방법을 제시하고 있다. ESK는 파일 파편을 바이트 배열로 보고 해당 배열 내에서 파일 헤더와 같은 파일 시그니처 패턴이 있는 지를 검색하는 역할을 한다. ESA는 행렬형태의 ESK를 효율적으로 계산하기 위하여 입력된 파일 파편내의 부분 바이트 배열이다. 파일의 헤더와 같은 시그니처 패턴을 파일 파편에서 찾아내는 것은 컴퓨터 포렌식 관점에서 매우 의미 있는 작업이며, 암호화되거나 압축된 파일 파편에서도 이러한 정보를 추출할 수 있으려면 본 논문에서 제시하고 있는 추가적인 연구가 필요하다.

이 외에도 *B. Li*외 2인의 추가적인 연구[5] 및 *Oliver de Vel*의 연구[6] 등에서 파일 파편 분석과 관련된 연구를 제시하고 있으나 주로 파일 파편은 압축되거나 암호화 되어 있지 않으며, 원본 파일을 재구성할 수 있는 파일 파편이 모두 존재한다는 가정을 가지고 파일 파편 분석 방법을 제시한 경우가 많다. 현재까지 제시된 파일 파편 분석에 관한 연구 및 실험 결과를 향상시키기 위하여 본 논문에서는 암호화 혹은 압축되어 있는 파일 파편을 구별하고, 각각에 맞는 처리를 통하여 압축되지 않은 평문을 획득하는 방법에 대한 연구 결과를 서술한다. 이는 압축이나 암호화로 인하여 손상된 데이터의 평문적 특성이나 통계적 특성을 복구하는 작업으로서 파일 파편의 분석에 있어서 필수적인 작업이다.

## Ⅲ. 파일 파편 분석 절차

컴퓨터 포렌식에서는 압수한 디스크를 조사 및 분석하는 것이 필수적이다. 이 때 컴퓨터 포렌식 수사관은 압수한 디스크의 할당 영역과 비할당 영역을 조사해야 하며 구체적인 절차는 [그림 1]과 같다.

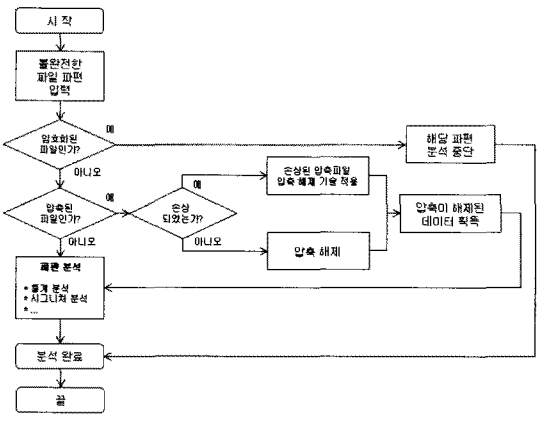



〔그림 1〕 디스크 이미지 분석

할당 영역에 정상적으로 존재하는 데이터는 응용 프로그램을 이용하여 실제 파일의 내용 및 메타 데이터(Meta Data) 등을 살펴보고, 손상된 데이터는 데이터 복구 도구 등을 이용하여 파일을 정상적인 형태로 복구한 다음 그 내용을 조사할 수 있다. 또한 각 파일의 형식(Format)이나 응용 프로그램이 가진 특징을 이용하여 은닉된 파일이 존재하는지를 탐색하고 은닉된 데이터를 추출하여 조사 및 분석해야 한다.

비할당 영역에 존재하는 데이터를 조사할 때는 EnCase[7]와 같은 포렌식 조사 도구를 이용하여 완전한 파일로서 존재하는 파일을 우선적으로 복구하여 추출해야 한다. 이러한 데이터는 삭제되었으나 다른 파일로 덮여 쓰이지 않은 데이터로서 할당 영역에 정상적으로 존재하는 파일을 분석하는 방법과 별도의 파일 카빙 기법을 사용하여 완전한 파일로 재구성이 가능하다. 이러한 데이터들을 제외하고 비할당 영역에 남아 있는 데이터는 완전한 파일로 재구성 할 수 없는 '불완전한 파일 파편'으로 정의한다. 즉, '불완전한 파일 파편'은 단독으로 파일 일부로서 존재하는 데이터이며 본 논문에서 제시하는 파일 파편을 분석하는 알고리즘은 다음 [그림 2]와 같다.

즉, 비할당 영역에서 획득한 파일 파편에 대하여, 암호화 여부와 압축 여부를 판단한 다음, 암호화된 경우는 손상된 형태의 암호문이기 때문에 의미 있는 정보를 추출할 수 없으므로 분석 대상에서 제외한다. 압축된 데이터의 경우 손상되지 않았다면 적용된 압축 알고리즘만 알면 압축을 풀 수 있다. 만약 획득한 파일 파편이 손상된 형태의 압축 데이터인 경우는 손상된 압축 파일의 압축을 해제하는 기술을 적용하여 압축 해제를 시도한다. 이러한 처리를 통해 압축이나 암호화가 되지 않은 평문을 획득할 수 있다면 통계적 특징 분석이나 시그니처 분석 등의 기존 연구 방법으로 파일 파편을 분석할 수 있다. 만약 압축이나 암호화가 되지 않은 평문을 획득할 수 없다면 입력으로 받은 파일 파편은 그 자체로서는 분석이 거의 불가능하다고 볼 수 있다. 이런 경우는 서로 연관이 있다고 판단되는 파일 파편들을 모아서 압축 및 암호화 판별을 하고, 동일한 처리를 해볼 수 있다. 다음 장에서 앞서 기술한 압축된 데이터와 암호화된 데이터를 구분하는 방법, 그리고 손상된 압축 파일 파편의 압축을 해제하는 기술에 대하여 기술한다.




[그림 2] 파일 파편 분석 알고리즘

## Ⅳ. 파일 파편 분석에 필요한 기술

[그림 1]에 제시된 불완전한 파일 파편은 비할당 영역에 존재하는 완전한 파일로 복원할 수 없는 파일 파편으로서 이러한 파일 파편은 적용된 파일 파편 분석 방법에 따라 다음과 같이 세 가지로 분류될 수 있다.

1) 암호화된 불완전한 파일 파편
2) 압축된 불완전한 파일 파편
3) 암호화 및 압축이 되지 않은 불완전한 파일 파편

일반적으로 이러한 파일 파편의 분석은 파일 파편이 가지고 있는 통계적인 특성 등을 이용하여 원본 파일을 추정하는 방식으로 이루어진다. 하지만 암호화나 압축과 같은 작업은 데이터의 엔트로피를 높임으로써 파일 파편이 원본 파일이 가졌던 것과 동일한 통계적 특성을 가지지 못하게 한다. 그러므로 암호화나 압축이 된 파일 파편은 일반적인 방법으로는 분석하기가 어렵다. 또한 암호화 된 데이터와 압축 된 데이터는 이진 값의 배열(Binary value sequence)만 보고서는 해당 데이터가 압축되었는지, 암호화되었는지를 판단하기 힘들다. 만약, 엔트로피가 높은 임의의 파일 파편을 수집했을 때 압축되었는지 암호화되었는지를 판단할 수 있다면 각각의 경우에 맞는 압축 해제 시도를 하거나, 암호 공격을 하거나 혹은 분석이 불가능한 데이터인지를 분류할 수 있을 것이며 이는 신속한 분석이 요구되는 컴퓨터 포렌식 수사에서 많은 시행착오를 줄여주는 역할을 한다.

### 4.1 암호화된 데이터와 압축된 데이터의 구분 기술

암호화 된 데이터와 압축된 데이터는 그 것의 Raw 데

이터 (Raw Data)를 읽는 것만으로는 구분할 수 없다. 본 논문에서는 NIST에서 제공하는 통계 테스트[8]를 이용하여 암호문과 압축된 데이터를 구분하는 방법을 제시한다. NIST에서 제공하는 통계 테스트는 암호 알고리즘의 난수성 검정을 수행하기 위한 테스트로서 미국의 AES 프로젝트를 수행한 NIST에서 SP800-22 문서(A Statistical Test Suite for Random and Pseudorandom Number Generators for Cryptographic Applications)를 기반으로 하고 있다[9]. 이 난수성 검정법은 블록암호 알고리즘의 난수성 검정을 위해 개발되었으며 가장 널리 사용되고 있는 난수성 검정법이다.

NIST의 16가지의 통계 테스트 다음 [표 1]과 같다. '*한양대학교 자연과학연구소*'[10]에서는 16가지의 테스트 중 Frequency Test, Serial Test, Poker Test 그리고 Runs distribution test에 대하여 실험에 요구되는 최소 비트수를 각각 512비트, 512비트, 512비트, 1024비트로 결정짓고 있다. 또한 각 실험에 따른 비트수에 대한 p-값을 구했을 때 전체적으로 평문에 대한 유의확률(p-value)이 암호문에 대한 유의확률(p-value)보다 안정적이라는 것을 각 파일 종류별로 보이고 있다. 이는 1종 오류를 범할 최대의 허용 기준인 유의 수준(α)을 '0.01'로 설정했을 때 검정통계량의 관측 값으로 귀무가설을 기각하려 할 때, 요구되는 유의수준의 최소값인 유의확률이 '0.01'보다 작으면 해당 데이터가 난수성을 가지지 않음을 판단한다. 해당 연구에서는 PDF, cpp, doc, hwp, ps, tex 등의 몇몇 파일을 제외한 데이터 군에서 평문의 유의 확률이 '0.01'보다 작은 범위에서 대부분 같은 영역에 존재한다는 결과를 제시하고 있다.

〔표 1〕 NIST 16가지 통계 테스트

| NIST 16가지 통계 테스트 | |
|---|---|
| 1 | Frequency Test |
| 2 | Frequency Test within a Block |
| 3 | Run Test |
| 4 | Test for the Longest Run of Ones in a Block Test |
| 5 | Binary Matrix Rank Test |
| 6 | Discrete Fourier Transform Test (DFT Test) |
| 7 | Non-overlapping Template Matching Test |
| 8 | Overlapping Template Matching Test |
| 9 | Maurer's 'Universal Statistical' Test |
| 10 | Lempel-Ziv Compression Test |
| 11 | Linear Complexity Test |
| 12 | Serial Test |
| 13 | Approximate Entropy Test |
| 14 | Cumulative Sum(Cusum) Test |
| 15 | Random Excursions Test |
| 16 | Random Excursions Variant Test |

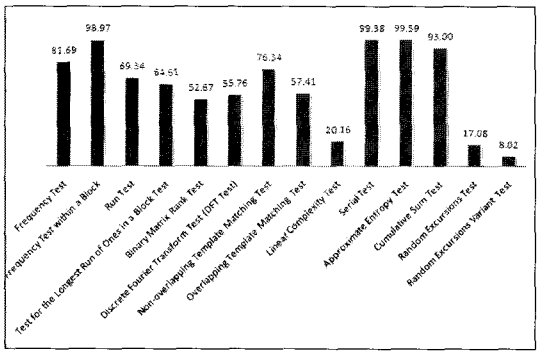
본 논문에서는 가장 일반적인 클러스터 크기인 4096 바이트 단위로 수집한 불완전한 파일 파편에 대하여 NIST의 16가지 통계 테스트를 수행함으로써 암호문, 압축문 혹은 암호문도 아니고 압축문도 아닌 데이터인지를 판단한다. 압축된 평문과 압축되지 않은 평문은 각 테스트를 이용한 유의 확률의 통계 값을 산출해 냄으로써 일부 가능하다. 그 일례로 [표 2]에서는 Frequency Test를 시행하였을 때 압축된 평문과 압축되지 않은 평문에 대한 유의 확률에 대한 통계 값을 제시하고 있다. 압축되지 않은 형식의 파일은 압축된 형식의 파일과 유의 확률의 차이를 나타내고 있다. 이러한 유의 확률의 차는 실험 데이터가 작을수록 더욱 뚜렷함을 확인할 수 있다.

따라서 유의 확률 값을 각 파일 종류별 유의 확률 대표값과 비교함으로써 일부 압축된 평문과 압축되지 않은 평문의 구별이 가능하다. 본 논문은 압축된 데이터 파편과 암호화된 데이터 파편을 구분하여 압축된 데이터 파편에 압축 해제를 시도하는 것을 목적으로 하고 있으며 따라서 압축된 데이터와 암호화된 데이터를 구분하는 방법에 대해 주목한다.

우선 각 파일 파편의 부분 수열이 난수성을 갖는 수열 즉, 암호화된 수열이라는 귀무가설을 검정하기 위한 유의 수준을 '0.01'로 설정하였다. 이 때 암호문을 평문으로 판단하는 확률은 거의 없음을 고려할 때 즉, 1종 오류는 거의 없음을 고려할 때-실제로 Random Excursions Variable Test를 제외한 나머지 15개의 테스트의 경우 90% 이상으로 암호문을 정확하게 판단함-압축된 평문을 암호문으로 판단하는 확률이 가장 작은

〔표 2〕 Frequency Test에 의한 평문의 유의 확률

| | 유의 확률 | |
|---|---|---|
| | 압축되지 않은 데이터 | 압축된 데이터 |
| 128 | 0.0123 | 0.2578 |
| 256 | 0.0019 | 0.4382 |
| 512 | 0.0000 | 0.3010 |
| 1024 | 0.0002 | 0.0361 |
| 2048 | 0.0000 | 0.0052 |




〔그림 3〕 임의의 크기를 가진 데이터에 시행한 NIST 통계 테스트 14개 압축문 판별력(단위 : Percentage)

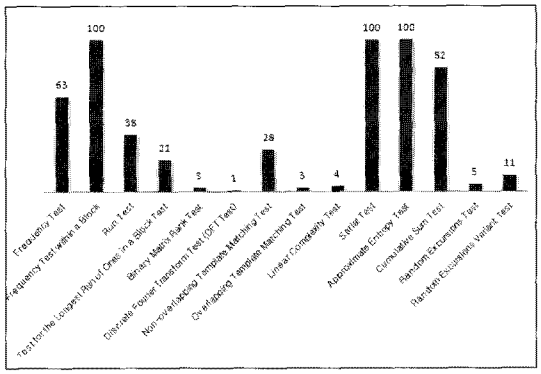
〔표 3〕 통계 테스트에 사용한 실험군에 대한 설명

| | 크기 | 개수 | 특징 |
|---|---|---|---|
| 실험군 | 임의 크기 | 486 | 암호화된 데이터나 압축된 데이터는 헤더나 메타 정보가 존재하지 않는 순수(Raw) 암호화된 데이터 및 압축된 데이터를 사용하였으며 이는 압축되거나 암호화되지 않은 헤더나 메타 정보로 인하여 압축 및 암호화에 의한 통계적 특성이 잘못 해석되는 경우를 배제하기 위함이다. |

테스트를 고르는 것이 중요한 작업이다. 따라서 387840 바이트 이상의 데이터로 실험이 가능한 Maurer's 'Universal Statistical' Test와 100000바이트 이상의 데이터로 실험이 가능한 Lempel-Ziv Compression Test를 제외하고, 임의의 크기를 가진 압축된 데이터 486개에 대하여 나머지 14개의 통계 테스트를 시행하였으며 자세한 실험군 선정과 테스트 결과는 [표 3] 및 [그림 3]과 같다.

[그림 3]에서 90% 이상의 확률로 압축문을 정확하게 판별해 내는 테스트는 Serial Test, Approximate Entropy Test, Frequency within a block Test, Cumulative Sum Test이며 이 4가지의 테스트가 임의의 크기를 가지는 파일에서 2종 오류가 가장 작다고 판단할 수 있다. 일반적으로 데이터의 크기에 따라 통계 테스트의 결과가 다를 수 있으므로 일반적인 클러스터 단위인 4096 바이트 크기의 파일에서도 위의 네 가지 테스트의 판별력이 높은지 판단할 필요가 있으며 그 결과는 [그림 4]와 같다.

[그림 4]에서 4096 바이트 크기를 가진 데이터 1000개에 대해 14개의 테스트를 시행한 결과를 나타내었다. 이 때, 14개의 통계 테스트에서 [그림 3]의 결과와 비교하면 14개 테스트에서 전체적으로 2종 오류율이 늘어




〔그림 4〕 4096 바이트 크기를 가진 데이터에 시행한 NIST 통계 테스트 14개 압축문 판별력 (단위 : Percentage, 소수점 반올림)

〔표 4〕 NIST 16가지 통계 테스트

| Test 명 | Test 시행 평균 시간(단위:초) |
|---|---|
| Frequency within a block Test | 0.00095 |
| Serial Test | 0.00123 |
| Approximate Entropy Test | 0.00278 |
| Cumulative Sum Test | 0.00031 |

났지만 Frequency within a block Test, Serial Test, Approximate Entropy Test 그리고 Cumulative Sum Test는 여전히 높은 압축문 판별력을 가지고 있다. 4096 바이트의 데이터에 대해 네 가지의 테스트를 시행한 평균 시간은 다음 [표 4]와 같다. 시간 측정은 4096 바이트 크기를 가진 데이터 1000개에 대하여 네 가지 테스트를 시행할 때 프로시저가 각각의 테스트 함수에 진입할 때와 함수에서 벗어날 때의 시간을 측정하여 평균값을 계산한 것이다.

즉, 네 가지 테스트 모두 클러스터 단위의 데이터의 종류를 판단하기에, 컴퓨터 포렌식 수사에 무리를 줄 만큼의 오랜 시간이 요구되지 않는다. 만약 30GB의 비할당 영역에서 압축되거나 암호화된 불완전한 데이터 파편이 20GB로 존재하고, 클러스터 단위가 4096 바이트라면 이러한 20GB에 대한 암호화 여부 혹은 압축 여부 판단 시간은 다음과 같이 약 7시간 18분여가 소요된다.

$$(20GB/4096Bytes)\times(0.00095\text{sec}+0.00123\text{sec}+0.00278\text{sec}+0.00031\text{sec})=26214.4\text{sec}$$

위의 네 가지 테스트의 결과를 바탕으로 난수성을 가진 데이터인지의 여부를 판단하는 알고리즘은 다음 [그

```
If( Frequency within a block Test   == Random   Or
    Serial Test                     == Random   Or
    Approximate Entropy Test        == Random   Or
    Cumulative Sum Test             == Random)
{
        Random;
}else
{
        Not-Random;
}
```

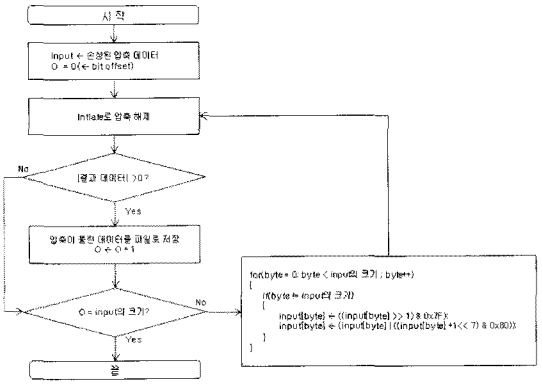
〔그림 5〕 3가지 테스트 결과 해석 알고리즘

림 5]와 같다. 이는 앞서 언급한 1종 오류 발생의 희귀성을 근거로 네 가지 테스트 결과를 종합적으로 판단한 것이다. 즉, 하나의 테스트라도 랜덤 데이터라고 판단했다면 해당파일 파편은 암호화 된 데이터라고 판단하는 것이다.

### 4.2 손상된 압축 파편의 압축 해제 기술

암호화되지 않았으나 상대적으로 엔트로피가 큰 데이터는 압축된 데이터라고 판단할 수 있으며 이 때, 압축된 데이터로 결정된 파일 파편은 압축 해제를 시도해 볼 수 있다. 압축된 파일 파편이 손상된 형태가 아니라면 압축을 해제하는 것이 가능하다. 하지만 압축된 파일 파편이 손상되면 압축을 정상적으로 해제하는 것은 매우 어려운 일이다. 일반적으로 데이터가 덮여 쓰이면 파일의 첫 부분이 존재하는 클러스터의 첫 부분부터 덮여 쓰이게 되는데 일반적으로 압축 데이터의 첫 부분이 없거나 손상되면 압축을 해제하는 것이 불가능하다고 알려져 있다. 이러한 특징은 데이터에 적용된 압축 알고리즘의 특징에 기인하는데, 본 논문에서는 가장 보편적으로 사용되는 압축 알고리즘인 Deflate 압축 알고리즘을 대상으로 손상된 압축 파편의 압축 해제 방법에 대하여 논의한다.

Deflate 압축 알고리즘은 LZ77과 허프만 코딩(Huffman Coding)을 사용하여 데이터를 압축한다. Deflate 압축 알고리즘으로 압축을 하는 것은 3가지 방법이 있는데 압축을 하지 않는 방법, LZ77과 고정 허프만 테이블(Fixed Huffman Table)을 이용한 방법이 있고 LZ77과 동적 허프만 테이블(Dynamic Huffman Table)을 새로 구성하여 코딩하는 방법이 있다. 압축할 데이터의 크기가 큰 경우 압축된 데이터는 여러 개의 블록의 형태를 띠며 개개의 블록은 압축할 때 사용자가 주는 설정 값에 따라 독립적일 수도, 종속적일 수도 있

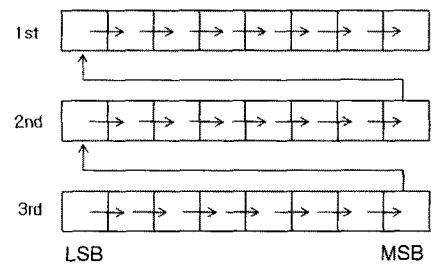



〔그림 6〕 손상된 압축 데이터 압축 해제 알고리즘

다. Deflate 압축 알고리즘으로 압축된 데이터의 압축을 해제하는 알고리즘을 Inflate 알고리즘이라 한다. 압축을 해제할 때는 허프만 코드를 읽어 들여 그 것을 LZ77 코드로 해석한 다음 LZ77 디코딩(Decoding)하는 방식으로 진행된다. 고정 허프만 테이블을 사용한 경우, 알고리즘 내에 저장되어 있는 테이블을 이용하여 디코딩하고 동적 허프만 테이블을 사용한 경우 데이터 내에 존재하는 테이블을 읽어 EOB(End Of Block)을 만날 때 까지 디코딩한다[11].

압축된 데이터가 첫 부분부터 존재한다면 압축을 풀 수 있다. 하지만 압축된 데이터의 첫 부분이 존재하지 않는다면 압축된 데이터의 압축을 푸는 것은 매우 어렵다. 왜냐하면 Deflate 압축 알고리즘에서 사용하는 LZ77 코딩 방법이 이전에 출현한 문자열의 특징을 이용하여 데이터를 압축하기 때문에 압축을 풀 때도 이전의 데이터를 참조하여 압축을 풀어야 하기 때문이다. 하지만 LSB에 맞추어 비트를 하나씩 제거해가며 압축 해제 알고리즘을 적용한다면 압축이 풀리는 경우가 존재한다. 이는 비트가 제거되면서 새로운 독립적인 압축 블록을 만났을 때, 혹은 Deflate 압축 알고리즘 내에서 고정 허프만 테이블을 이용하여 압축되었을 경우에서 허프만 디코딩(Decoding) 과정에서 나타나는 LZ77의 Distance 코드가 해당 데이터를 읽은 크기를 벗어나지 않는다면 압축 해제가 가능하다. [그림 6]에서는 앞부분이 존재하지 않는 압축된 데이터의 압축을 해제하는 방법을 간략하게 제시하고 있다.

손상된 압축 데이터 형태의 파일 파편의 압축을 해제하기 위해서는 LSB 순서에 맞춰 전체 데이터를 매번 1비트 쉬프트(Shift)하면서 Inflate를 이용하여 압축을 해제해야 한다. 쉬프트 연산을 하는 방법은 다음 [그림 7]




〔그림 7〕 Bit-by-Bit Shift

과 같다. 즉, Inflate가 데이터를 읽어 들이는 순서를 맞추기 위한 작업으로 이는 한 비트를 제거하고 난 후에도 Inflate 알고리즘이 인식하는 데이터의 순서가 최초의 데이터와 같아야 한다.

비트 단위로 쉬프트 연산을 한 각 파일마다 Inflate 압축 해제 알고리즘으로 압축을 해제하여 압축 해제된 데이터의 길이가 '0'이 아니라면 파일로 저장한다. 이때 압축이 풀린 데이터 중에는 실제로 압축이 되기 전 상태의 파일 파편이 있을 수도 있고 우연히 Inflate 압축 알고리즘에 따라 엉뚱한 값이 채워져 있을 수 있다.

대부분의 경우 가장 큰 크기를 가지고 있는 데이터가 포렌식 수사관들이 파일 파편으로부터 유의미한 정보를 획득할 수 있는 '압축이 풀린 데이터'라고 할 수 있다. 이러한 방법으로 압축이 풀리는 경우를 다음 [표 5]와 같다. 이 실험 결과는 Deflate로 압축된 데이터에만 해당하는 것이다. 다른 압축 알고리즘을 사용한 경우는 압축을 해제할 수 있는 방법에 대한 연구가 추가로 필요하다.

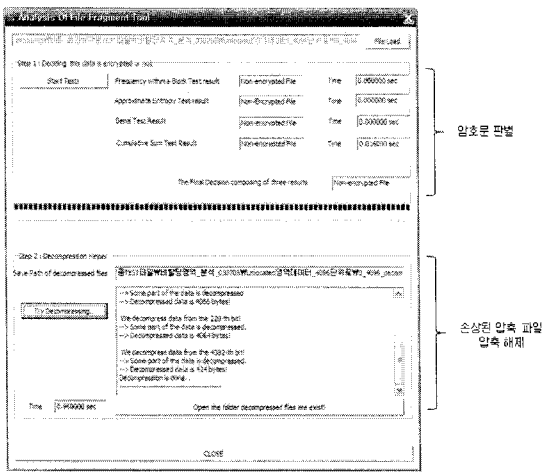
이러한 압축 해제 기술을 적용하여 압축을 풀 수 있다면 현재 분석이 불가능하다고 판단되어 왔던 많은 데이터 파편에서 유익한 정보를 추출해 낼 수 있다. 본 논문에서 제시한 압축 해제 기술은 기존의 손상된 압축 파일을 복구하는 데 쓰였던 기술과는 다른 방법론을 적용하고 있다. 기존의 압축 파일 도구는 해당 압축 파일의 시그니처와 Raw 압축 데이터에의 오프셋(offset) 정보가 온전히 남아 있을 경우에 주로 복구 가능하며 이는 파일 내에 존재하는 파일 구성에 필수적인 메타 데이터를 이용하여 데이터를 복구하는 것이다. 따라서 Raw 데이터가 손상되었을 경우에는 각종 메타 데이터가 존재하여 파일을 재구성하였을지라도 실제 파일 내용을 압축을 풀어서 보여줄 수는 없다. 기존의 손상된

〔표 5〕 비트별 압축 해제로 압축이 풀리는 경우와 풀리지 않는 경우

| 단일 블록 여부 (single/multiple) | 압축 블록의 순서 (block index) | 압축 방법 (1:압축하지 않음, 2:LZ77&FixedHuffman, 3:LZ77&DynamicHuffman) | 압축 블록의 첫 부분에 존재하는 LZ77 코드의 특징 | 해당 압축 블록 해제 가능 여부 |
|---|---|---|---|---|
| | | | | |
| Single | (첫 번째) | 1 | | 가능 |
| Single | (첫 번째) | 2 | | 가능 |
| Single | (첫 번째) | 2 | | 가능 |
| Single | (첫 번째) | 3 | | 가능 |
| Multiple | 첫 번째 | 1 | | 가능 |
| Multiple | 첫 번째 | 2 | | 가능 |
| Multiple | 첫 번째 | 3 | | 가능 |
| Multiple | 첫 번째 아님 | 1 | | 가능 |
| Multiple | 첫 번째 아님 | 2 | 최대 거리 코드가 현재 블록 첫 부분까지의 오프셋보다 작음 | 가능 |
| Multiple | 첫 번째 아님 | 2 | 최대 거리 코드가 현재 블록 첫 부분까지의 오프셋보다 큼 | 불가능 |
| Multiple | 첫 번째 아님 | 3 | 최대 거리 코드가 현재 블록 첫 부분까지의 오프셋보다 작음 | 가능 |
| Multiple | 첫 번째 아님 | 3 | 최대 거리 코드가 현재 블록 첫 부분까지의 오프셋보다 큼 | 불가능 |

〔표 6〕 기존의 복구 도구와 논문에서 제시하는 압축 해제 방법의 비교

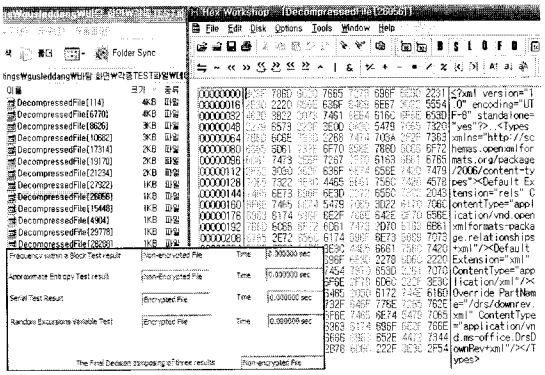
| | | 기존의 복구 도구 | 논문에서 제시한 방법 |
|---|---|---|---|
| 메타 데이터 없이 Raw 압축 데이터만 있을 때, 압축을 해제할 수 있는가? | | 불가능 | 가능 |
| 다수의 압축 블록으로 이루어진 Raw 압축 데이터에서 첫 번째 블록이 손상되었을 때, 압축을 해제할 수 있는가? | | 불가능 | 가능 |
| 단일 압축 블록의 경우, 블록의 첫 부분이 손상되었을 때, 압축을 해제할 수 있는가? | LZ77&고정허프만 코딩을 사용하여 압축한 경우 | 불가능 | 때때로 가능 |
| | LZ77&동적허프만 코딩을 사용하여 압축한 경우 | 불가능 | 불가능 |




〔그림 8〕 구현된 파일 파편 분석 프로그램화면

압축 파일 복구 도구와 본 논문에서 제시한 방법에 대한 비교를 다음 [표 6]에서 보이고 있다.

[그림 8]은 암호화된 데이터와 압축된 데이터를 구분하는 기술과, 손상된 압축된 형태의 파일 파편을 분석하는 기술을 하나의 도구로 구현한 것이다. Step1에서 압축되었거나 암호화되었다고 판단되는 파일을 입력으로 받고 네 가지의 통계 테스트를 이용하여 암호화 여부를 판단한 후, 압축된 평문이라면 Step2에서 압축 해제를 시도한다. 본 프로그램은 LSB 순서로 한 비트씩 제거해가며 매번 압축을 해제하게 되며 4096바이트의 크기를 가진 데이터의 경우 약 0.0065초가 소요된다. 사용자가 압축 해제된 데이터를 저장한 폴더를 선택하면 압축을 푼 데이터의 크기가 0 바이트보다 큰 압축 해제된 데이터가 저장되는데 걸리는 시간이다. 데이터의 압축 해제가 가능한 경우였다면 일반적으로 압축 해제된 파일의 크기가 가장 큰 것이 정상적으로 압축이 풀린 것으로 해석할 수 있다. 만약 20GB에 대하여 암호화 및 압축 여부를 판단하고 4096 바이트 단위로 압축 해제를



〔그림 9〕 논문에서 제시한 방법으로 획득한 텍스트

시도한다면 다음과 같이 약 14시간 30분 정도의 시간이 소요될 수 있다. 하지만 전체 디스크에 대하여 테스트를 하는 경우는 극히 드문 일이며 분석이 불가하다고 판단한 파편에 대하여만 본 테스트와 압축 해제를 시도할 것이므로 실제로는 훨씬 적은 시간이 소요될 것으로 예상된다.

$$(0.005(\text{4개의 테스트 소요 시간}) + 0.0065(\text{압축 해제 및 저장시간}))*(20GB/4096Byte) = \text{약 } 14.56 \text{ 시간}$$

디스크의 비할당 영역에 존재하는 데이터 파편 중 몇 개를 추출하여 본 도구로 압축을 해제해 본 결과 텍스트를 비롯한 각종 유의미한 정보를 추출할 수 있었다. 일례로 다음 [그림 9]은 xml의 일부를 획득한 화면이다. 비할당 영역에 존재하는 데이터를 추출하여 암호화되지 않은 압축 파일이라는 것을 확인한 후 해당 데이터의 압축 해제를 시도하였다. '26056번째 LSB부터 압축이 풀린 일부 파일에서 이 파편이 xml 파일의 부분이라는 것을 알 수 있고 xml을 분석한 결과 이는 Office Open XML 형식을 사용하는 MS Office 2007에서 생성된 문서를 구성하는 xml 파일 중 하나라 볼 수 있으며 해당

클러스터와 연속된 몇몇 클러스터를 이어서 압축을 풀어보면 좀 더 많은 부분을 획득할 수 있으리라는 것을 추측할 수 있다. 동일한 과정으로 많은 데이터 파편을 분석한 결과 상당한 텍스트 데이터나 특정 파일 포맷상 텍스트가 존재하는 데이터를 획득할 수 있었다.

## V. 결 론

본 논문에서는 비할당 영역에 존재하는 파일 파편을 분석함에 있어서 생길 수 있는 기술적인 어려움을 해결하고자 파일 파편을 분석하는 데 있어서 압축 데이터와 암호화 된 데이터를 통계 테스트를 이용하여 판별하는 방법, 그리고 Deflate 압축 알고리즘으로 압축된 손상된 압축 파편의 압축을 해당 압축 알고리즘의 특성을 이용하여 해제할 수 있는 방법을 제시하였다. 즉, 비할당 영역에서 파일 파편을 획득했다면 본 논문에서 제시한 방법을 이용하여 해당파일 파편의 압축 및 암호화 여부를 판단하고 각각의 파편의 특성에 맞는 분석 방법을 적용해야 한다. 압축된 파일 파편이 압축 데이터의 첫 부분부터 존재하는 경우에는 압축을 해제하여 압축되지 않은 상태의 데이터를 획득하기 쉽지만 그렇지 않은 경우에는 압축을 해제하는 것이 매우 어렵다. 이러한 경우에는 본 논문에서 제시한 방법으로 압축 해제를 시도해 볼 수 있다. 기존에 불완전한 파일 파편에 대해 조사 및 분석하는 방법이 제시된 적이 없기 때문에 본 논문에서 제시한 방법은 컴퓨터 포렌식 수사에 매우 의미하는 바가 크다.

기존의 컴퓨터 포렌식 영역은 할당 영역에 존재하는 데이터와 비할당 영역에 존재하는 완전한 파일로의 복구가 가능한 파일 파편에 대한 방법론에 대한 연구가 주로 이루어지고 있다. 그러므로 차후에는 파일 파편을 분석하는 알고리즘을 기존의 컴퓨터 포렌식 수사 절차와 통합하여 하나의 '파일 파편 분석' 영역을 추가하여 좀 더 새롭고 구체적인 디스크 포렌식 수사 모델을 제시할 것이다. 이는 파일 카빙의 영역을 더욱 확장할 수 있는데 특히 '압축' 옵션을 선택하여 데이터를 디스크에 저장하면 Deflate 압축 알고리즘을 이용하여 데이터 영역을 압축하게 되는데 이러한 경우 비할당영역에 존재하는 거의 모든 데이터는 압축되어 있으며 이의 압축 여부를 판단하고 압축을 푸는 것은 파일 카빙 영역에서도 필수적인 작업이 된다. 따라서 비할당 영역에 존재하는 파일 파편을 분석할 때는 체계적인 방법론을 이용하여 접근해야 하며 분석된 파편의 파일 카빙에의 활용성도 염두에 두어야 한다. 이와 같은 이유로 본 논문에서 제시한 데이터 파편 분석 및 압축 해제 방법은 컴퓨터 포렌식에서의 디스크 분석 영역에서 체계화되어야 하는 필수적인 요소이다. 차후에는 파일 카빙 영역과 본 논문에서 제시한 영역을 연계시켜 데이터 파편 분석을 통한 파일 카빙에 대한 새로운 방법론 및 기술을 제시하고 Deflate 압축 데이터뿐만 아니라 다른 압축 알고리즘을 사용하여 압축된 손상된 데이터의 압축을 해제하는 방법에 대하여 제시할 것이다.

## 참고문헌

[1] Louretta, Good Practice Guide for computer based Electronics Evidence version 3.0, Association of Chief Police Officers, 2003

[2] Kulesh Shanmugasundaram, Nasir Memon, "Automatic Reassembly of Document Fragments via Context Based Statistical Models," *Proceedings of the 19th Annual Computer SecurityApplications Conference (ACSAC)*. Las Vegas, NV, USA, 2003, pp. 152-159.

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

[4] Binglong Li, Qingxian Wang, and Junyong Luo, "Forensic Analysis of Document fragment based on SVM", 2006 International Conference on Intelligent Information Hiding and Multimedia Signal Processing(IIH-MSP'06), pp.236-239

[5] Binglong Li, Qingxian Wang, Junyong Luo, et al. "Classifying model of Document Fragments and the Research" of its Key Issue[J]. *Journal of Harbin Institute of Technology*, 2006, pp. 834-839.

[6] Oliver de Vel, "Augmented sequence spectrum kernels for semi-structured document categorization", *Proceedings of the Workshop on Text Mining and Link Analysis, 18th International Joint Conference on Artificial Intelligence*

*(IJCAI)*, 2003, pp. 213-221

[7] www.guidancesoftware.com

[8] Andrew Rukhin, Juan Soto, James Nechvatal, Miles Smid, Elaine Barker, Stefan Leigh, Mark Levenson, Mark Vangel, David Banks, Alan Heckert, James Dray, San Vo "A STATII STIICAL TEST SUIITE FOR RANDOM AND PSEUDORANDOM NUMBER GENERATORS FOR CRYPTOGRAPHIIC APPLIICATIIONS", NIST Special Publication 800-22, (with revisions dated May 15, 2001)

[9] N. I. of Standards and Technology, Special publication 800-22: A Statistical Test Suite for Random and Pseudo-random Number Generators for Cryptographic Applications. http://scrc.nist.gov/rng /rng.pdf, 2000.

[10] 한양대학교 자연과학연구소, 암호통신문 탐지 기술 연구 및 S/W 개발, 한국 정보보호센터, 1999년 12월

[11] D. Salomon, Data Compression - The Complete Reference - 4th edition, Spinger-Verlag, 2007

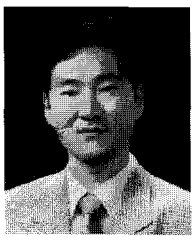
## 〈著者紹介〉

**박 보 라 (Bora Park) 학생회원**

2007년 2월 : 부산대학교 수학교육과 학사

2007년 3월~ : 고려대학교 정보경영공학전문대학원 석사과정

<관심분야> 디지털 포렌식, 정보 은닉 이론



**이 상 진 (Sangjin Lee) 종신회원**

1989년 2월~1999년 2월 : 한국전자통신연구원 선임 연구원,

1999년 2월~2001년 8월 : 고려대학교 자연과학대학 조교수,

2001년 9월~현재 : 고려대학교 정보경영공학전문대학원 교수

<관심분야> 대칭키 암호, 정보은닉이론, 컴퓨터 포렌식