The Journal of The Institute of Internet, Broadcasting and Communication (IIBC)
Vol. 21, No. 4, pp.9-13, Aug. 31, 2021. pISSN 2289-0238, eISSN 2289-0246

https://doi.org/10.7236/JIIBC.2021.21.4.9
JIIBC 2021-4-2

디지털 포렌식을 위한 SHA-256 활용 데이터 수정 감지시스템 제안

A Proposal on Data Modification Detection System using SHA-256 in Digital Forensics

장은진*, 신승중**

Eun-Jin Jang*, Seung-Jung Shin**

요 약 통신 기술의 발달과 더불어 다양한 형태의 디지털 범죄가 증가하고 있고, 이에 따라 디지털 포렌식에 대한 필요성이 높아지고 있다. 더욱이 특정인이 중요한 데이터를 담고 있는 텍스트 문서를 고의적으로 삭제하거나 수정할 경우 데이터 수정 감지 여부를 확인하는 시스템을 통해 특정인과 범죄와의 연관성을 입증할 수 있는 중요한 자료가 될 수 있을 것이다. 이에 본 논문은 텍스트 파일을 중심으로 암호화 기법 중 하나인 SHA-256의 hash data, 생성 시간, 수정 시간, 접근 시간 및 파일의 크기 등을 분석하여 대상 텍스트 파일의 수정 여부를 비교할 수 있는 데이터 수정 감지 시스템을 제안한다.

Abstract With the development of communication technology, various forms of digital crime are increasing, and the need for digital forensics is increasing. Moreover, if a textual document containing sensitive data is deliberately deleted or modified by a particular person, it could be important data to prove its connection to a particular person and crime through a system that checks for data modification detection. This paper proposes a data modification detection system that can analyze the hash data, file size, file creation date, file modification date, file access date, etc. of SHA-256, one of the encryption techniques, focusing on text files, to compare whether the target text file is modified or not.

Key Words : Digital Forensic, Analyze Data, SHA-256, Hash Data, Encryption

Ⅰ. 서 론

최근 꾸준하게 이슈 되고 있는 블록체인의 핵심기술로 암호화 기술의 중요성이 대두되고 있다. 이 중 데이터 전처리를 통한 암호화 연산으로 원본데이터로의 복호화가 불가능한 SHA-256 암호화 기법이 사용되고 있다.

또한, 정보 통신 기술의 발달로 디지털 범죄율이 증가하고 있다. 개인의 신분을 쉽게 감출 수 있고, 원하는 시간 동안 다양한 방식으로의 데이터 조작이 가능한 온라인 환경에서의 범죄는 날이 갈수록 지능화되며 증가하고 있다.

그림 1.은 최근 5년간 국내 디지털 범죄 발생 건수를

*정회원, 한세대학교 IT융합학과
**종신회원, 한세대학교 IT융합학과(교신저자)
접수일자 2021년 6월 23일, 수정완료 2021년 7월 23일
게재확정일자 2021년 8월 6일

Received: 23 June, 2021 / Revised: 23 July, 2021 /
Accepted: 6 August, 2021
*Corresponding Author: dmswls1061@naver.com
Dept of IT Convergence, Hansei University, Korea

나타낸다. 2015년 110,109건이었던 디지털 범죄는 2019년 180,499건을 기록하며 꾸준히 증가하는 추세를 보이고 있다.

이렇듯 디지털 범죄에 대항하기 위한 각종 증거 수집 및 데이터 복구를 위한 디지털 포렌식의 필요성이 나날이 증가하고 있지만, 현행법상 지정된 수사 범위에서만 증거를 수집해야 하는 등 증거 수집에 대한 여러 가지 제약조건이 존재한다.

이러한 상황에서 새로운 증거 수집 방법으로 특정인이 중요한 데이터에 대한 고의적인 조작이 확인된다면 특정인과 범죄와의 연관성을 입증할 수 있는 중요한 자료가 될 수 있을 것으로 기대된다. 이에 본 논문은 SHA-256의 hash data, 텍스트 파일의 크기, 생성 시간, 수정 시간 등의 분석하여 대상 파일의 수정 여부를 비교할 수 있는 데이터 수정 감지시스템을 제안하고자 한다.

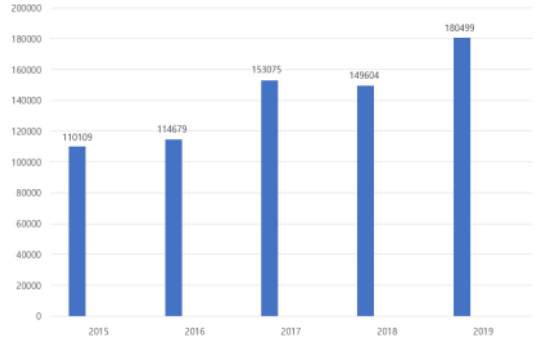

그림 1. 최근 5년간 디지털 범죄 발생 건수(출처:경찰청)
Fig. 1. Number of Digital Crimes committed in the last five years(Source: Police Department)

II. 파일 시스템 및 SHA-256 암호화

1. 파일 시스템

파일 시스템은 컴퓨터에서 저장되는 자료들을 하나의 객체로 취급하여 사용자가 쉽게 사용할 수 있도록 관리하는 방식을 말한다. 다시 말해 파일의 이름을 정하고 저장 및 검색을 위해 논리적으로 어떠한 위치에 배치시켜야 하는지에 대한 방법을 구성한 시스템이라고 할 수 있다[1].

이러한 파일 시스템의 특성을 이용해 특정 파일을 분석하고, 접근하는 것이 가능해진다. 대표적으로 많이 사용되는 파일 시스템으로 FAT(File Allocation Table)형식이 있다.

FAT 파일 시스템의 구조는 Reserved Area, FAT Area, Data Area로 나뉜다.

Reserved Area는 예약된 영역으로서 이 안에 파일의 첫 번째 시작인 Boot Record를 비롯하여 FSINFO(File System Information), Boot Record Backup, FSINFO Backup, Boot Strap, Reserved Sector 등으로 구성된다. FAT Area는 FAT1과 FAT2로 구성되는데, FAT1은 파일 또는 디렉터리에 할당 유무가 기록되는 부분으로 4byte의 공간이 필요하다. Data Area는 실제 파일 데이터가 저장되는 부분이다.

그림 2.는 FAT파일 시스템의 구조에 대해 나타낸다.

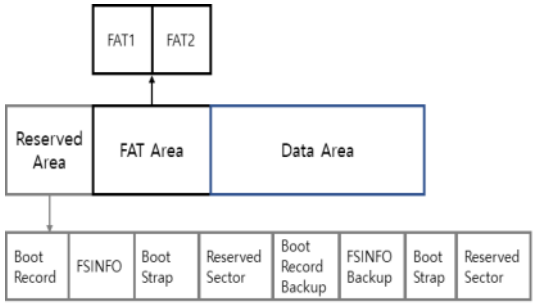

그림 2. FAT 파일 시스템의 구조
Fig. 2. Structure of FAT File System

2. SHA-256 암호화

SHA는 Secure Hash Algorithm의 약자로 미국 국가안보국(NSA)에서 설계한 암호 hash 함수이다. SHA 함수에는 SHA-1, SHA-224, SHA-256, SHA-384, SHA-512, SHA-512/224, SHA-3 등이 있다. 표 1.은 주요 SHA hash 함수의 종류 및 특징을 나타낸다.

표 1. SHA hash 함수의 종류 및 특징
Table 1. Types and Features of SHA Functions

함수 명	특징
SHA-1	SHA-0 함수에 비트 회전 연산 추가
SHA-224	SHA-256의 256비트에서 32비트를 줄임
SHA-256	256비트로 64자리 문자열 반환
SHA-384	SHA-1 의 대체 함수 중 하나
SHA-512	512비트 hash 값 생성
SHA-3	SHA-1과 SHA-2 의 대체 함수

이 중 SHA-256은 보안성이 높아 가장 널리 사용 되고 있는 암호화 hash 함수이다. SHA-256의 hash 결과는 256bit로서 원본 자료를 암호화 하여 전처리하고, 전

처리 된 자료를 바탕으로 hash를 계산하게 된다.

자료를 암호화 하고, hash값을 생성하는 과정에서 SHA-256은 원본 자료의 일부 데이터만을 저장하고 있기 때문에 암호화 된 hash값만을 가지고 원본데이터로의 복호화는 불가능하다. 따라서 비트코인의 핵심기술인 블록체인에서 대표적으로 사용되고 있다.

메시지 전처리 과정에서 원본 데이터의 bit 길이가 512의 배수가 되도록 padding을 추가한다.

메시지 전처리 과정은 다음과 같이 진행된다. 첫째, 원본 메시지의 바로 뒤에 비트 '1'을 하나 추가한다. 둘째, 메시지의 길이가 512의 배수가 되도록 메시지에 0을 추가한다. 메시지의 마지막 64bit에는 원본 메시지의 bit 길이를 적는다[2].

그림 3.은 SHA-256 메시지 전처리 과정을 나타낸다.

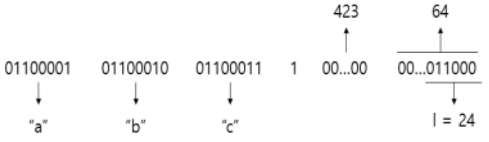

그림 3. SHA-256 메시지 전처리 과정
Fig. 3. SHA-256 Message Preprocessing Process

전처리된 메시지는 해싱 단계를 거치게 된다. 전처리된 512의 배수 형태의 메시지를 512bit 단위로 나누어서 여러 개의 chunk로 분할한다. 분할 된 chunk 들은 차례로 그림 4.와 같은 Compression Function 연산을 통해 최종 hash값을 얻게 된다.

그림 4.는 SHA-256 Compression Function 연산 방법을 나타낸다.

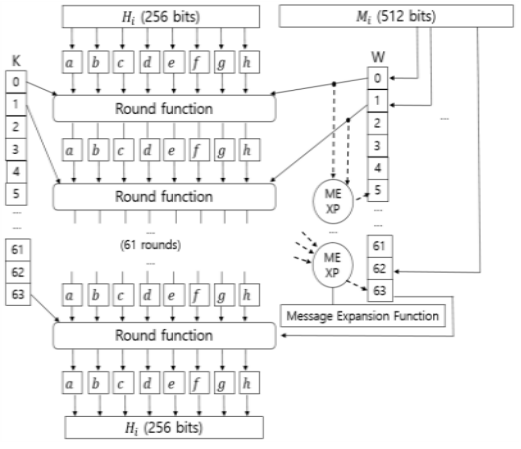

그림 4. SHA-256 Compression Function 방법
Fig. 4. Method of SHA-256 Compression Function

Ⅲ. 데이터 수정 감지시스템 제안

1. 데이터 수정 감지시스템 제안

본 논문에서는 디지털 포렌식을 위한 새로운 증거 수집 방법으로 SHA-256 활용 데이터 수정 감지시스템을 제안한다. 본 시스템을 활용할 경우 텍스트 파일에 대한 수정 등의 조작 여부를 가시적인 데이터로 확인할 수 있고, 이러한 데이터를 활용할 경우 특정인과 범죄와의 연관성을 입증할 수 있는 중요한 자료가 될 수 있을 것으로 기대된다.

본 시스템은 암호화 방식인 SHA-256 hash data를 확인하고, 텍스트 파일의 생성 시간, 수정 시간, 접근시간, 파일 크기 등을 비교 분석하여 파일의 조작 여부를 확인한다.

텍스트 파일 내의 내용이 수정되었다면 고유 값을 갖는 SHA-256 hash data 역시 변경되고, 파일의 크기 또한 변하기 때문에 특정인의 임의적인 데이터 조작 여부를 확인할 수 있다.

특히, 텍스트 파일의 내용을 수정하는 경우 외에도 단지 여백을 저장하여 원본 데이터와 차이가 없어 보이는 상황에서도 원본 데이터와 수정된 데이터 간의 차이점을 확인할 수 있다.

그림 5.는 본 논문에서 제안되는 데이터 수정 감지시스템에 대한 구현도를 나타낸다.

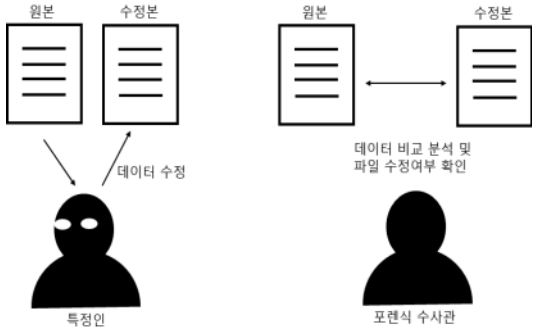

그림 5. 데이터 수정 감지시스템 구현도
Fig. 5. Data Modification Detection System Implementation Diagram

2. 시스템 개발환경

본 논문에서 제안되는 데이터 수정 감지 시스템의 개발 환경은 다음과 같다.

OS 환경은 Window 64bit이고, 개발도구는 PyCharm을 사용하였으며, 개발 언어는 Python3.9를 사용하였다.

3. 시스템 구현

본 시스템에서 원본 파일의 수정 여부를 확인하기 위해 원본 파일의 일부 내용을 임의로 조작한 후 별도로 파일을 저장하여, 파일의 SHA-256 hash data와 파일 생성 시간, 수정 시간, 엑세스 시간 및 파일 크기를 비교한다.

원본 파일의 파일명은 test로 하고, 수정된 파일의 파일명은 1로 한다. 원본 파일의 텍스트 내용을 일부 수정하여 수정된 파일로 저장하였다. 그림 6.은 원본 파일과 수정된 텍스트 파일의 모습을 나타낸다.

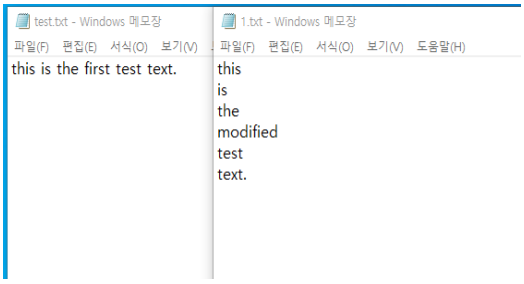

그림 6. 원본 파일(좌)과 수정된 파일(우)
Fig. 6. Original Files(Left) and Modified Files(Right)

두 파일의 정보를 비교하기 위해 hashlib 라이브러리를 사용하여 코드를 작성한다. 또한, 파일 생성 시간 등을 확인하기 위해 datetime 라이브러리를 사용하며, 파일의 사이즈를 확인하기 위해 getsize 라이브러리를 사용한다.

표 2. 수정 감지 시스템 코드
Table 2. Modification Detection System Code

```
import hashlib
import os
import datetime
from os.path import getsize

filename = "test.txt"
basename=os.path.basename(filename)
file_name=os.path.splitext(basename)[0]
f=open('test.txt', 'rb')
data = f.read()
f.close()
print(filename+' 의 SHA-256: ' +
hashlib.sha256(data).hexdigest())
ctime =
os.path.getctime('C:/Users/eunjin/PycharmProjects/pythonP
roject/test.txt')
mtime =
os.path.getmtime('C:/Users/eunjin/PycharmProjects/python
Project/test.txt')
atime =
os.path.getatime('C:/Users/eunjin/PycharmProjects/pythonP
roject/test.txt')
size=getsize('C:/Users/eunjin/PycharmProjects/pythonProjec
t/test.txt')
print(str(filename),'파일 만든 시간 ',
datetime.datetime.fromtimestamp(ctime))
print(str(filename),'파일 수정 시간 ',
datetime.datetime.fromtimestamp(mtime))
print(str(filename),'마지막 엑세스 시간 ',
datetime.datetime.fromtimestamp(atime))
print(str(filename),'파일 크기 ', size)
print('\n')

filename = "1.txt"

basename=os.path.basename(filename)
file_name=os.path.splitext(basename)[0]

f=open('1.txt', 'rb')
data = f.read()
f.close()

print(filename+' 의 SHA-256: ' +
hashlib.sha256(data).hexdigest())
ctime =
os.path.getctime('C:/Users/eunjin/PycharmProjects/pythonP
roject/1.txt')
mtime =
os.path.getmtime('C:/Users/eunjin/PycharmProjects/python
Project/1.txt')
atime =
os.path.getatime('C:/Users/eunjin/PycharmProjects/pythonP
roject/1.txt')
size=getsize('C:/Users/eunjin/PycharmProjects/pythonProjec
t/1.txt')

print(str(filename),'파일 만든 시간 ',
datetime.datetime.fromtimestamp(ctime))
print(str(filename),'파일 수정 시간 ',
datetime.datetime.fromtimestamp(mtime))
print(str(filename),'마지막 엑세스 시간 ',
datetime.datetime.fromtimestamp(atime))
print(str(filename),'파일 크기 ', size)
```

표 2.는 본 논문에서 제안되는 수정 감지 시스템에 대한 코드를 나타내고, 그림 7.은 수정 감지 시스템의 분석 결과를 나타낸다.

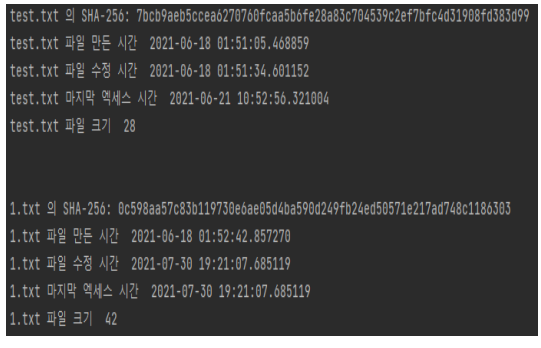

그림 7. 수정 감지 시스템 분석 결과
Fig. 7. Modification Detection System Analysis Results

그림 6.에서 확인되는 바와 같이 원본과 수정된 파일의 내용이 일부 변경되어 있었다. 이러한 두 개의 텍스트 파일을 본 시스템을 통해 확인해보면 그림 7.과 같은 결과를 확인할 수 있다.

그림 7. 과 같이 파일의 SHA-256값이 바뀌었고, 파일을 만든 시간, 수정 시간 및 마지막 액세스 시간이 차이가 나는 것을 확인할 수 있었다. 또한, 파일 크기가 원본 파일은 28이고, 수정된 파일은 42로 확인되었다. 파일 크기 비교 기능을 활용할 경우 공란 입력 후 저장 등과 같이 미세한 수정에 대한 수치 차이를 확인할 수 있었고, 이를 통해 파일의 수정 여부를 가시적으로 확인할 수 있게 되었다.

Ⅳ. 결 론

본 논문에서는 디지털 포렌식 분석을 위해 SHA-256 hash data와 파일 수정 시간 및 크기를 비교하여 텍스트 파일의 수정 여부를 확인할 수 있는 데이터 수정 감시 시스템을 제안하였다.

본 논문에서 제안되는 데이터 수정 감시 시스템을 활용할 경우 데이터 hash data, 파일 수정 시간 및 파일 크기의 비교를 통해 데이터의 임의 조작 여부를 확인할 수 있을 것으로 기대되며, 특정인과의 범죄 혐의점을 입증할 수 있는 가시적인 증거 자료가 될 수 있을 것으로 기대된다.

향후 증거 채택을 위하여 분석 기록 등록 기능 추가 및 손상 데이터 복구 방법 등에 대한 연구가 진행되어야 할 것이다.

References

[1] https://bmind305.tistory.com/42

[2] http://www.secmem.org/blog/2019/07/21/sha256/

[3] Ji-Yoon Ham, Joshua I. James, "A Feature Comparison of Modern Digital Forensic Imaging Software", The Journal of The Institute of Internet, Broadcasting and Communication, Vol. 19, No. 6, pp. 15-20, Dec 2019. DOI: https://doi.org/10.7236/JIIBC.2019.19.6.15

[4] Jae-Jeong Hwang, "Forensic Detection of Filtration Forgery in Digital Images", The Journal of The Institute of Electronics and Information Engineers, Vol.56, No.1, Jan 2019. DOI: https://doi.org/10.5573/ieie.2019.56.1.85

[5] Eun-Jin Jang, Seung-Jung Shin, "ProPosal of New Data Processing Function to Improve the Security of Self-driving Cars' Systems", The Journal of The Institute of Internet, Broadcasting and Communication, Vol. 20, No.46, pp. 81-86, Aug 2020. DOI: https://doi.org/10.7236/JIIBC.2020.20.4.81

[6] Do-Hyun Kim, Jun-Ki Kim, Sang-Jin Lee, "An Analysis of Google Data from a Digital Forensic Perspective", The Journal of the Korea Institute of Information and Communication Engineering, Vol. 24, No.12, pp. 1662-1669, Dec 2020. DOI:http://doi.org/10.6109/jkiice.2020.24.12.1662

[7] Se-Yool Park, Sang-Jin Lee, "Forensic Investigation of HWP File", The Journal of Digital Forensics, Vol.14, No.4, pp. 408-425, Dec 2020. DOI:10.22798/KDFS.2020.14.4.408

[8] Yeon-Joo Lee, Jeong-Min Kim, Sung-Jin Lee, "A Study of Polaris Office Forensic Artifact", The Journal of Digital Forensics, Vol.14, No.4, pp. 368-378, Dec 2020. DOI:10.22798/KDFS.2020.14.4.368

저 자 소 개

장 은 진(정회원)

- 2012년 2월 : 단국대학교(학사) 식량생명공학과, 중국어학과
- 2019년 2월 : 한세대학교 대학원 IT융합학과 (공학석사)

신 승 중(종신회원)

- 1988년 : 세종대학교 대학원 경영학과졸업(석사)
- 1994년 : 건국대학교 대학원 전자계산학과 졸업(석사)
- 2000년 : 국민대 대학원 정보관리학과 졸업(박사)
- 1995년 ~ 2003년 : 중부대학교 정보보호학과 교수
- 2003년 ~ 현재 : 한세대학교 ICT융합학과 교수
- 주 관심분야 : 정보보호, 이동통신, 통신공학