Journal of The Korea Institute of Information Security & Cryptology ISSN 1598-3986(Print)
VOL.33, NO.1, Feb. 2023 ISSN 2288-2715(Online)
https://doi.org/10.13089/JKIISC.2023.33.1.51

윈도우 환경에서 카카오톡 데이터 복호화 및 아티팩트 분석 연구*

조 민 욱,[1†] 장 남 수[2‡]
[1,2]세종사이버대학교 (대학원생, 교수)

Study on The Data Decryption and Artifacts Analysis of KakaoTalk in Windows Environment*

Minuook Jo,[1†] Nam Su Chang[2‡]
[1,2]Sejong Cyber University (Graduate student, Professor)

요 약

카카오톡, 라인, 페이스북 메신저 등과 같은 메신저는 누구나 사용하는 범용적인 의사소통 수단이다. 사용자들에게 제공되는 편의 기능과 사용 시간이 증가할수록 아티팩트 안에 남게 되는 사용자의 행위 정보 또한 증가하고 있으며, 이는 디지털 포렌식 수사 관점에서 중요한 증거로 활용되고 있다. 그러나 보안상의 이유로 현재 대부분의 데이터는 암호화되어 저장되고 있다. 또한, 의도적인 조작, 은닉, 삭제 등의 은폐 행위가 증가하여 디지털 포렌식 분석 시간이 지연되는 문제를 야기하고 있다.

본 논문에서는 국내에서 가장 많은 사용자를 가진 메신저인 카카오톡에 대해 Windows 환경에서 데이터 복호화 및 아티팩트 분석 방안에 관한 연구를 수행하였다. 효율적인 복호화 키 획득 방안, 삭제 시도한 메시지 식별 및 복호화 방안을 제시하고 썸네일 아티팩트를 분석한다.

ABSTRACT

Messengers such as KakaoTalk, LINE, and Facebook Messenger are universal means of communication used by anyone. As the convenience functions provided to users and their usage time increase, so does the user behavior information remaining in the artifacts, which is being used as important evidence from the perspective of digital forensic investigation. However, for security reasons, most of the data is currently stored encrypted. In addition, cover-up behaviors such as intentional manipulation, concealment, and deletion are increasing, causing the problem of delaying digital forensic analysis time.

In this paper, we conducted a study on the data decryption and artifacts analysis in a Windows environment for KakaoTalk, the messenger with the largest number of users in Korea. An efficient way of obtaining a decryption key and a method of identifying and decrypting messages attempted to be deleted are presented, and thumbnail artifacts are analyzed.

Keywords: Digital Forensics, Messenger, KakaoTalk, Brute Force, Decryption

Received(09. 27. 2022), Modified(12. 12. 2022), Accepted(12. 12. 2022)
* 본 논문은 2022년도 한국정보보호학회 하계학술대회에 발표한 우수논문을 개선 및 확장한 것임

† 주저자, jominuook@gmail.com
‡ 교신저자, nschang@sjcu.ac.kr(Corresponding author)

I. 서 론

메신저란 인터넷과 같은 네트워크를 통해 다수의 사용자가 실시간으로 의사소통할 수 있는 프로그램이다. 일반적으로 의사소통에 사용하던 전화나 문자에 비해 다자간 대화, 음성 통화, 영상 통화 및 대용량 파일 전송 등 훨씬 더 많은 기능을 제공한다. 스마트폰의 출시와 이동 통신 기술의 발달로 메신저에 대한 접근성이 향상되면서 대중적인 의사소통 수단으로 자리 잡았다.

그러나 메신저가 제공해주는 편의 기능이 각종 범죄에 악용되는 경우도 많이 발생하고 있다. 대중적으로 알려진 "n번방 성 착취물 제작 및 유포 사건"과 같은 불법 촬영물 유포나 각종 피싱 범죄, 기밀자료 유출 등이 그 사례이다. 직접적인 범죄 수단으로 활용되지 않더라도 메신저는 그 특성상 사용자의 다양한 행위 정보들이 남게 되어 수사에서 중요한 증거로 활용되기도 한다. 실제로 "16개월 입양아 학대 사망 사건"이나 "가평계곡 살인 사건" 등 여러 사건에서 대화 기록은 중요한 증거로 활용되었다.

위와 같은 중요성에도 불구하고 대다수 메신저가 보안상의 이유로 각종 데이터를 암호화하여 저장하고 있으며, 용의자가 의도적으로 메시지를 삭제하는 등 안티 포렌식 행위는 증가하여 디지털 포렌식 수사는 계속 어려워지고 있다. 또한, 최근 출시되는 스마트폰은 암호화를 기본적으로 수행하게 되어 있으므로 기존의 JTAG나 Chip-Off와 같은 방법으로는 더 이상 유의미한 데이터를 확보할 수 없게 되었다. 이로 인해 기기의 암호를 해제하지 못하여 중요한 증거를 확보하지 못하는 사례들도 많아지고 있다[1-3].

이에 대응하여 스마트폰 포렌식 방안 연구가 다수 진행되었다. 그러나 USIM을 활용한 압수수색 방안에서 카카오톡 대화 기록을 획득하기 위해서는 카카오 계정 로그인이 필요하였으며[1], 제조사 백업 앱 기반 데이터 획득 기법에서는 카카오톡 대화 기록을 획득할 수 없었다[2]. [3]에서는 지문정보를 이용한 스마트폰 포렌식 방안을 제시하였으나, 생체정보를 활용한 포렌식 방안은 아직 법제화되지 않은 상태이다.

본 논문에서는 Windows 환경의 카카오톡에서 데이터 복호화 및 아티팩트 분석 방안에 관한 연구를 수행하였다. 복호화 시 사용되는 입력 파라미터인 userId를 빠르게 획득할 수 있는 효율적인 전수조사 방안을 제시하여 영장청구 제한시간 내에 분석을 위한 충분한 시간을 확보할 수 있도록 하였다. 메시지 삭제 시도 여부를 식별하고 모든 대화 상대에게서 삭제 시도한 메시지 복호화에 성공하여 용의자의 의도적인 안티 포렌식 행위에 대응할 수 있도록 하였다. 썸네일 아티팩트 및 사용자 행위에 따른 썸네일의 변화를 분석하여 각종 불법 촬영물 유포나 기밀자료 유출 등의 범죄에 대응할 수 있도록 하였다.

본 논문의 구성은 다음과 같다. 2장에서는 메신저에 관해 이루어진 기존의 복호화, 삭제된 메시지 복구, 썸네일에 대한 연구에 관해 기술한다. 3장에서는 복호화 시 사용되는 입력 파라미터인 userId를 빠르게 획득할 수 있는 효율적인 전수조사 방안, 삭제 시도한 메시지 식별 및 복호화 방안을 제시하고 썸네일 아티팩트 및 사용자 행위에 따른 썸네일의 변화를 분석한다. 마지막으로 4장에서 결론을 내리며 마무리한다.

II. 기존 연구 결과

2.1 복호화 방안[4]

Windows 환경에서 카카오톡은 대화 기록 데이터베이스 파일 내에 수발신 된 모든 메시지를 저장한다. 해당 데이터베이스 파일은 대화방 별로 각각 "%LocalAppData%\Kakao\KakaoTalk\users\{userDir}\chat_data\chatLogs_{chatId}.edb" 형식으로 암호화하여 생성한다. 이를 복호화하는 방안은 [4]에서 제시되었다. AES 알고리즘을 사용하여 페이지 크기인 4,096바이트 별로 복호화하며, 페이지 크기가 16바이트의 배수이므로 별도의 패딩은 사용되지 않는다. 구체적인 알고리즘은 Fig. 1.과 같다.

```
Input:  key, iv, encDB
Output: decDB

1: decDB ← []
2: i ← 0
3: while (i < encDB.length)
4:     decDB ← decDB ||
           AES128/CBC/NOPADDING_DEC
           (encDB[i:i+4096], key, iv)
5:     i ← i + 4096
6: return decDB
```

Fig. 1. Method for decrypting a chatLogs database

Fig. 1.과 같이 대화 기록 데이터베이스를 복호화하기 위해서는 key, iv 쌍을 생성해야 한다. pragma와 userId를 512바이트가 될 때까지 반복하여 연접한 후 MD5 해싱을 수행하면 key가 된다. key에 대해 Base64 인코딩 수행 후 MD5 해싱을 수행하면 iv가 된다. 구체적인 알고리즘은 Fig. 2.와 같다.

Fig. 2.의 입력 파라미터인 pragma와 userId는 각종 데이터를 암호화하는 데 흔히 사용된다. pragma 생성 방안은 다음과 같다. 하드 코딩된 바이트 배열을 key로 하여, UUID와 ModelName, SerialNumber를 연접한 결과에 대해 AES 암호화를 수행한다. 위 결과에 대해 SHA512 해싱 수행 후 Base64 인코딩을 수행하면 pragma가 생성된다. 이때 사용되는 입력 파라미터들은 모두 기기 내에서 획득할 수 있는 정보들이다. UUID는 Windows 운영체제 정보에서 획득할 수 있으며, ModelName과 SerialNumber는 하드디스크 정보에서 획득할 수 있다. 구체적인 알고리즘은 Fig. 3.과 같다.

Input: pragma, userId
Output: key, iv

```
key ← pragma || userId
while (key.length < 512)
    key ← key || key
key ← key[0:512]
key ← MD5(key.getBytes())
iv ← MD5(Base64.encode(key))
return key, iv
```

Fig. 2. Method for generating key and iv for a chatLogs database

Input: UUID, ModelName, SerialNumber
Output: pragma

```
key ← hardcoded byte array
iv ← 0
pragma ← String.format("%s|%s|%s",
        UUID, ModelName, SerialNumber)
pragma ← AES128/CBC/PKCS#7_ENC
        (pragma.getBytes(), key, iv)
return Base64.encodeToString
        (SHA512(pragma))
```

Fig. 3. Method for generating a pragma

반면, Fig. 2.의 입력 파라미터 중 userId는 카카오톡 회원가입 시 사용자별로 부여되는 시퀀스로 기기 내에서 획득할 수 없는 정보이다. 따라서 userId를 획득하기 위한 효율적인 전수조사 방안에 관한 연구가 필요하다.

2.2 기존 삭제된 메시지 복구 방안

메신저 사용 중 남겨진 대화 기록이 수사에서 중요한 증거로 활용되는 경우가 많아지면서 이를 은닉하기 위해 의도적으로 메시지를 삭제하는 안티 포렌식 행위가 증가하고 있다. 이에 대응하여 메시지 삭제 여부를 식별하고 삭제된 메시지를 복구하기 위한 연구가 다수 진행되었다.

[5]에서는 삭제된 오버플로우 데이터를 오버플로우 페이지에서 복구하는 방안을 제시하였다. 실제로 iOS 환경에서 획득한 카카오톡 데이터베이스 파일을 이용한 실험에서 삭제된 오버플로우 데이터 복구에 성공하였다. 그러나 카카오톡 사용 중에 실험되지 않았으며, 오버플로우 페이지는 언제든지 새로운 레코드로 덮어 쓰일 수 있다.

[6-7]에서는 SQLite 데이터베이스를 사용하는 메신저에서 저널 파일을 통한 삭제된 메시지 복구 방안을 제시하였다. Android 환경의 카카오톡에서 사용자가 메시지 삭제 시 값이 0으로 변경되어 메시지 삭제 여부를 식별할 수 있었으며, 대화방을 나간 경우에는 저널 파일 내 잔존 데이터를 이용하여 삭제된 메시지를 복구할 수 있었다. 그러나 저널 파일은 임시 파일로 삭제가 빈번하게 일어나기 때문에 단기간 내에 삭제된 메시지만을 포함하고 있다.

[1]에서는 용의자가 메신저에서 수행한 대화 기록을 확보하기 위한 압수수색 방안으로 용의자의 USIM을 다른 기기에 삽입하는 방안을 제시하였다. 이를 통해 카카오톡에서 나에게서만 삭제한 메시지를 포함하여 최근 3일간 이루어진 대화 기록을 획득하는 데 성공하였다. 그러나 카카오 계정 로그인이 필요하며 모든 대화 상대에게서 삭제한 메시지는 복구할 수 없었다.

기존 연구들은 단기간 내에 삭제된 일부 메시지만을 복구할 수 있었다.

2.3 기존 썸네일 아티팩트 분석 방안

메신저는 텍스트 메시지 외에도 사진, 동영상 등

미디어 형식 메시지를 전송할 수 있다. 미디어 파일은 상대적으로 용량이 크기 때문에 반복적인 다운로드 과정을 생략하고 사용자가 빠르게 인식할 수 있도록 썸네일을 사용자 기기 내에 미리 생성한다. 최근에는 사용자 기기 성능이 향상되며 미디어 파일을 축소하지 않고 원본 그대로 저장하기도 한다. 그러나 미디어 파일 전송 기능을 악용한 기밀 자료 유출, 성 착취물 유포 등 각종 범죄가 증가하고 있으며, 최근에는 불법 성적 촬영물 및 아동·청소년 성 착취물에 대한 소지, 시청 죄가 신설되었다. 썸네일은 원본 미디어 파일의 존재 사실을 입증하는 근거가 되며, 사용자 행위에 따른 썸네일의 변화를 분석하여 사용자의 소지, 유포, 시청 등 각종 행위를 입증할 수 있다. 이러한 썸네일의 위치를 식별하고 사용자 행위에 따른 썸네일의 변화를 분석하기 위한 연구가 다수 진행되었다.

[8-9]에서는 Android 환경에서 갤러리, 카메라, 클라우드, 메신저 애플리케이션들이 생성하는 사진, 동영상, 썸네일 아티팩트들의 위치를 식별하였다. 이를 기반으로 임의제출 확인서 작성 및 지정한 시간 내 생성된 아티팩트들을 수집하는 기능을 포함한 현장용 모바일 포렌식 도구를 구현하였다. 특히 카카오톡에서 생성하는 썸네일과 동영상 아티팩트의 위치를 식별하였다.

[10]에서는 Android 환경에서 텔레그램, 와이어, 디스코드 애플리케이션이 생성하는 사진, 동영상 메시지에 대한 캐시 파일의 위치와 특징을 분석하였다. 이렇게 생성된 캐시 파일을 근거로 디지털 성 착취물 시청 행위를 입증하는 방안을 제시하였다.

[11]에서는 Android 환경에서 인스타그램 애플리케이션이 생성하는 각종 아티팩트를 데이터베이스와 파일 관점에서 분석하였다. 사용자가 게시물 작성 과정에서 사진을 앱에 로드하면 cache 디렉토리에 복사되며, 게시물이 업로드되고 나면 files 디렉토리에 복사되는 사실을 밝혔다. 또한 사용자가 메인 피드에 접속하면 피드 화면 내 보여졌던 사진들이 images 디렉토리에 저장되며, 저장된 시간을 기반으로 메인 피드 접속 시간을 추정하는 방안을 제시하였다.

[12]에서는 Android 환경에서 갤러리 애플리케이션이 썸네일을 썸캐시라는 파일 내에 연속적으로 저장하는 특징을 밝혔다. 이를 기반으로 썸캐시 파일 내에 저장된 썸네일들을 추출하는 방안을 제시하고 썸네일 추출 도구를 구현하였다. 또한 사진 촬영, 열람, 삭제 등 사용자의 각종 행위에 따른 썸네일의 변화를 분석하였다.

III. 제안하는 카카오톡 데이터 복호화 및 아티팩트 분석 방안

3.1 분석 환경

본 논문은 Windows 환경의 카카오톡에서 데이터 복호화 및 아티팩트 분석 방안에 관한 연구를 수행하였다. 카카오톡 내에서 이루어지는 각종 암호화 및 복호화 과정을 분석하기 위해 정적 분석에는 IDA를 사용하였으며, 동적 분석에는 x64dbg와 Cheat Engine을 각각 디버깅과 메모리 스캔에 사용하였다. Java 및 IntelliJ IDEA를 복호화 실험에 사용하였으며, DB Browser for SQLite를 복호화한 데이터베이스 조회에 사용하였다. HxD Hex Editor를 각종 아티팩트의 데이터를 확인하기 위한 16진수 편집기로 사용하였다. 분석에 사용된 구체적인 환경 및 도구 정보는 Table 1.과 같다.

Table 1. Analysis environment and tools

Classification		Details
Analysis target	Processor	AMD Ryzen 7 3700X 8-Core Processor 3.60 GHz
	Memory	32GB
	Graphics card	NVIDIA GeForce GTX 1660
	Storage	Samsung SSD 850 PRO 256GB
	Operating system	Windows 10 Pro 21H2 19044.1645
	KakaoTalk	v3.3.9.3088
Analysis tools	IDA	v7.6
	x64dbg	vDec 5 2021, 15:31:39
	Cheat Engine	v7.3
	Java	v15.0.2
	IntelliJ IDEA	v2021.1.3
	DB Browser for SQLite	v3.12.1
	HxD Hex Editor	v2.5.0.0

3.2 제안하는 전수조사 방안

Fig. 1-3.에서 [4]에서 제안한 대화 기록 데이터베이스 복호화 방안을 소개하였다. 그러나 userId는 복호화 시 사용되는 입력 파라미터 중 유일하게 기기 내에서 획득할 수 없는 정보로 디지털 포렌식 수행 시 어려움이 존재한다. 본 절에서는 userId를 빠르게 획득할 수 있는 효율적인 전수조사 방안을 제시한다.

3.2.1 일반적인 전수조사 방안

userId를 획득하기 위한 일반적인 전수조사 방안은 복호화한 데이터베이스에서 무결성 검사 구문이 정상적으로 실행될 때까지 userId를 1씩 증가시키며 복호화를 반복 수행하는 것이다. 이를 정리하면 다음과 같다. pragma와 userId를 512바이트가 될 때까지 반복하여 연접한 후 MD5 해싱을 수행하면 key가 된다. key에 대해 Base64 인코딩 수행 후 MD5 해싱을 수행하면 iv가 된다. 생성한 key, iv 쌍을 이용하여 페이지 크기인 4,096바이트 별로 AES 복호화 수행 후 무결성 검사 구문 "pragma integrity_check"가 정상적으로 실행되면 올바른 userId 획득에 성공한 것이다. userId를 1씩 증가시키며 올바른 userId를 획득할 때까지 이를 Fig. 4.와 같이 반복 수행하면 된다.

그러나 위 방안은 데이터베이스 파일 크기만큼의 File I/O와 데이터베이스 연결 및 쿼리 실행이 필요하다. 이를 userId가 변경될 때마다 반복 수행하여 전수조사에 많은 시간이 소요된다.

```
Input:  pragma, encDB, maxUserId
Output: userId
 1: i ← 1
 2: while (i <= maxUserId)
 3:     key ← pragma || i
 4:     while (key.length < 512)
 5:         key ← key || key
 6:     key ← key[0:512]
 7:     key ← MD5(key.getBytes())
 8:     iv ← MD5(Base64.encode(key))
 9:     decDB ← []
10:     j ← 0
11:     while (j < encDB.length)
12:         decDB ← decDB ||
             AES128/CBC/NOPADDING_DEC
             (encDB[j:j+4096], key, iv)
13:         j ← j + 4096
14:     Files.write("temp.db", decDB)
15:     SQLite3_open("temp.db")
16:     try
17:         SQLite3_execute
            ("pragma integrity_check")
18:         return i
19:     catch Exception
20:         pass
21:     i ← i + 1
```

Fig. 4. Method for general brute force

3.2.2 헤더 기반 전수조사 방안

SQLite3 데이터베이스 파일의 상위 16바이트 헤더 포맷은 이미 알려져 있다. 데이터베이스 파일 상위 16바이트를 복호화한 결과와 알려진 헤더 포맷을 비교하는 방법으로 전수조사를 수행할 수 있다. 이를 통해 데이터베이스 파일 전체에서 상위 16바이트로 복호화 대상이 축소되며, 데이터베이스 연결 및 쿼리 실행이 생략되어 Fig. 5.와 같이 빠르게 전수조사 수행이 가능하다.

그러나 위 방안은 key, iv 쌍이 userId에 의존적이기 때문에 userId가 변경될 때마다 AES 키 스케줄을 반복하여 생성해야 하는 비효율적인 부분이 존재한다.

```
Input:  pragma, encDB, maxUserId
Output: userId
 1: header ← encDB[0:16]
 2: i ← 1
 3: while (i <= maxUserId)
 4:     key ← pragma || i
 5:     while (key.length < 512)
 6:         key ← key || key
 7:     key ← key[0:512]
 8:     key ← MD5(key.getBytes())
 9:     iv ← MD5(Base64.encode(key))
10:     result
          ← AES128/CBC/NOPADDING_DEC
            (header, key, iv)
11:     if (result == normalHeader)
12:         return i
13:     i ← i + 1
```

Fig. 5. Method for header-based brute force

3.2.3 사용자 디렉토리명 기반 전수조사 방안

본 논문에서는 위 방안들에 비해 효율적인 사용자 디렉토리명 기반 전수조사 방안을 제시한다. 카카오톡은 사용 중 다양한 아티팩트들을 자동으로 생성한다. 이때, 사용자와 관련된 아티팩트는 사용자별로 "%LocalAppData%\Kakao\KakaoTalk\users" 디렉토리 내 서로 다른 디렉토리를 생성하여 저장한다. 사용자 디렉토리명 생성 방안은 다음과 같다. "KAKAOTALK_PC_FOREVER" 문자열에 대해 MD5 해싱을 수행하면 key가 된다. key에 대해 Base64 인코딩 수행 후 MD5 해싱을 수행하면 iv가 된다. users 디렉토리의 절대 경로와 생성한 key, iv 쌍을 이용하여 userId에 대해 AES 암호화를 수행한 결과를 연접한다. 마지막으로 pragma에 대해 MD5 해싱을 수행하면 key가 되며, key에 대해 Base64 인코딩 수행 후 MD5 해싱을 수행하면 iv가 된다. 생성한 key, iv 쌍을 이용하여 위 결과에 대해 AES 암호화 수행 후 SHA1 해싱을 수행하면 사용자 디렉토리명이 생성된다. 구체적인 알고리즘은 Fig. 6.과 같다.

사용자 디렉토리명 생성 시 userId가 입력 파라미터로 사용되는 점을 이용하여 실제 사용자 디렉토리명과 userId를 달리 해가며 생성한 사용자 디렉토리명을 비교하는 방법으로 Fig. 7.과 같이 전수조사를 수행할 수 있다.

제안하는 방안에서는 File I/O와 데이터베이스 연결 및 쿼리 실행이 생략되며, 사용되는 모든 key, iv 쌍이 userId와 무관하여 AES 키 스케줄을 반복 사용할 수 있다. 또한 반복하여 암호화하는 문자열은 최초 1회만 암호화하도록 하여 효율성을 증대시켰다.

```
Input:  pragma, userDirHome, userId
Output: userDir

userIdStr ← String(userId)
key ← "KAKAOTALK_PC_FOREVER"
key ← MD5(key.getBytes())
iv ← MD5(Base64.encode(key))
result ← AES128/CBC/PKCS#7_ENC
         (userIdStr.getBytes(), key, iv)
result ← userDirHome || "\"
         || toHexString(result)
key ← MD5(pragma.getBytes())
iv ← MD5(Base64.encode(key))
result ← AES128/CBC/PKCS#7_ENC
         (result.getBytes(), key, iv)
return toHexString(SHA1(result))
```

Fig. 6. Method for generating a user directory name

```
Input:  pragma, userDirHome, userDir,
        maxUserId
Output: userId

userDirBytes ← strToHex(userDir)
key1 ← "KAKAOTALK_PC_FOREVER"
key1 ← MD5(key1.getBytes())
iv1 ← MD5(Base64.encode(key1))
key2 ← MD5(pragma.getBytes())
iv2 ← MD5(Base64.encode(key2))
prefix ← userDirHome[0:
         userDirHome.length / 16 * 16]
prefix
    ← AES128/CBC/NOPADDING_ENC
      (prefix.getBytes(), key2, iv2)
iv2 ← prefix[-16:]
userDirHome
    ← userDirHome[prefix.length:] || "\"
i ← 1
while (i <= maxUserId)
    userIdStr ← String(i)
    result ← AES128/CBC/PKCS#7_ENC
        (userIdStr.getBytes(), key1, iv1)
    result ← userDirHome
             || toHexString(result)
    result ← AES128/CBC/PKCS#7_ENC
        (result.getBytes(), key2, iv2)
    result ← SHA1(prefix || result)
    if (result == userDirBytes)
        return i
    i ← i + 1
```

Fig. 7. Method for brute force based on the user directory name

카카오톡 신규 회원가입 시 2022년 5월 9일 기준, 3억 8천 대의 userId가 발급되는 것을 확인하였다. 이를 고려하여 최대 사용자 수를 4억 명으로 가정한 알고리즘별 전수조사 소요 시간은 Table 2.와 같다. 이때, Fig. 4.의 소요 시간은 100만 건 당 소요 시간을 기준으로 추정하였다.

Android 또는 Windows 환경에서 카카오톡 대화 기록 추출을 지원하는 상용 도구들과 본 논문에서 제시하는 방안[Fig. 1-3, 7.]을 사용 시 필요한 사항들을 비교하면 Table 3.과 같다. 이때, Android 환경은 최신 스마트폰을 사용 중인 일반적인 환경을 가정하여 Android 12 버전의 루팅 되지 않은 상태를 기준으로 하였다.

Table 2. Time required for brute force by the algorithm

Input		
maxUserId	400,000,000	
Algorithm	Time required (hh:mm:ss)	Remarks
Fig. 4.	35:33:20	Estimation
Fig. 5.	00:17:50	
Fig. 7.	00:01:30	

Table 3. Comparison of requirements with commercial forensics tools

Program	Android	Windows
Magnet AXIOM	unlocked	email, password
MD-NEXT	unlocked	-
MREAPER	unlocked	-
proposed plan	-	pragma, user directory

3.3 제안하는 삭제 시도한 메시지 복호화 방안

메신저에서 메시지 삭제 여부를 식별하고 삭제된 메시지를 복구하기 위한 기존 연구들에서는 저널 파일이나 비 할당 영역을 이용한 방안이 주로 연구되었다. 그러나 저널 파일은 삭제가 빈번하게 일어나며, 비 할당 영역은 언제든지 덮어 쓰일 수 있어서 최근 삭제된 일부 메시지만을 확인할 수 있었다[5-7]. 특히 용의자의 USIM을 이용한 압수수색 방안에서는 모든 대화 상대에게서 삭제한 메시지는 복구할 수 없었다[1]. 본 절에서는 대화 기록 데이터베이스를 기반으로 메시지 삭제 시도 여부를 식별하고 모든 대화 상대에게서 삭제 시도한 메시지를 복호화하는 방안을 제시한다. 대화 기록 데이터베이스는 앞서 살펴본 바와 같이 Fig. 7.을 통해 userId를 빠르게 획득하여 Fig. 1-3.과 같이 복호화할 수 있다.

카카오톡에서 사용자가 메시지 삭제 시도 시 나에게서만 삭제와 모든 대화 상대에게서 삭제라는 2가지 옵션이 제공된다[Fig. 8.].

사용자가 나에게서만 삭제를 시도하면 deleted 컬럼 내 값이 1로 변경되어 이를 통해 메시지 삭제 시도 여부를 식별할 수 있으며, message 컬럼 내 값이 빈 값으로 수정되어 데이터베이스 레코드만으로

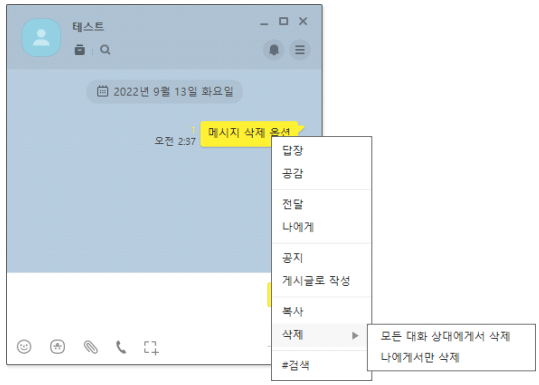

Fig. 8. Message deletion option

	logId	message	deleted	is_encrypt_delete_msg
1	2869489956622346240		1	0

Fig. 9. Message deleted for me

는 삭제 시도한 메시지를 복구할 수 없다[Fig. 9.].

사용자가 모든 대화 상대에게서 삭제를 시도하면 is_encrypt_delete_msg 컬럼 내 값이 1로 변경되어 이를 통해 메시지 삭제 시도 여부를 식별할 수 있으며, message 컬럼 내 값이 암호화된 값으로 수정되어 복호화 방안에 관한 연구를 통해 삭제 시도 후 암호화된 메시지를 복호화할 수 있다[Fig. 10.].

사용자가 모든 대화 상대에게서 삭제를 시도하여

	logId	message	deleted	is_encrypt_delete_msg
1	2869479119446779905	odrvlHKO…	0	1

Fig. 10. Message deleted for everyone

Input: encMsg
Output: decMsg

```
1: temp ← ""
2: i ← 0
3: while (i < encMsg.length)
4:     temp ← temp || encMsg[i]
5:     i ← i + 2
6: decMsg ← Base64.decode(temp)
7: key ← "==Talk=Delete=All=Message="
8: key ← MD5(key.getBytes())
9: iv ← MD5(Base64.encode(key))
10: decMsg ← AES128/CBC/PKCS#7_DEC
          (decMsg, key, iv)
11: return String(decMsg)
```

Fig. 11. Method for decrypting a message that attempted to delete

암호화된 메시지를 복호화하는 방안은 다음과 같다. “==Talk=Delete=All=Message=” 문자열에 대해 MD5 해싱을 수행하면 key가 된다. key에 대해 Base64 인코딩 수행 후 MD5 해싱을 수행하면 iv가 된다. 암호화된 메시지에서 짝수 번째 글자만을 추출하여 연접한 후 Base64 디코딩을 수행한 결과에 대해 앞서 생성한 key, iv 쌍을 이용하여 AES 복호화를 수행하면 메시지 원문을 획득할 수 있다. 구체적인 알고리즘은 Fig. 11.과 같다.

이는 카카오톡 서버가 아닌 사용자 기기 내 존재하는 대화 기록 데이터베이스 파일을 기반으로 하여 메시지 삭제 시도 후 경과 시간과 무관하게 메시지 삭제 시도 여부를 식별하고 복호화할 수 있다.

3.4 제안하는 썸네일 아티팩트 분석 방안

메신저는 상대적으로 용량이 큰 미디어 파일에 대해 썸네일을 미리 생성하여 반복적인 다운로드 과정을 생략하고 사용자가 빠르게 인식할 수 있도록 한다. 이렇게 생성된 썸네일을 통해 원본 미디어 파일의 존재 사실과 사용자 행위에 따른 썸네일의 변화를 분석하여 사용자의 각종 행위를 입증할 수 있다. 본 절에서는 썸네일이 저장되는 각 디렉토리를 분석하고 암호화된 썸네일을 복호화하는 방안을 제시한다. 대화 기록 데이터베이스를 기반으로 특정 메시지에 대한 썸네일을 식별하는 방안을 제시한다. 마지막으로 사용자 행위에 따른 썸네일의 변화를 분석하고 사용자 기기 내 저장된 원본 미디어 파일의 경로를 식별하는 방안을 제시한다.

카카오톡은 기기 내 썸네일 저장 시 유형에 따라 “%LocalAppData%\Kakao\KakaoTalk\users\{userDir}” 디렉토리 내 서로 다른 하위 디렉토리에 나누어 Table 4.와 같이 저장한다.

썸네일은 저장 시 파일명과 내용을 모두 암호화하여 “cng” 확장자로 저장하기 때문에 내용을 확인하기 위해서는 복호화 방안에 관한 연구가 필요하다. 암호화된 썸네일을 복호화하는 방안은 다음과 같다. pragma와 userId를 연접한 후 MD5 해싱을 수행하면 key가 된다. key에 대해 Base64 인코딩 수행 후 MD5 해싱을 수행하면 iv가 된다. 생성한 key, iv 쌍을 이용하여 암호화된 썸네일에 대해 AES 복호화를 수행하면 복호화된 썸네일을 획득할 수 있다. 구체적인 알고리즘은 Fig. 12.와 같다.

Table 4. Types of thumbnails saved by directory

Subdirectory	Thumbnail type
/chat_data/cli	Transmitted photos
/chat_data/cli/thumbnail	Transmitted photos/videos (miniaturized)
/chat_data/cli_http_v2	URL link Boards
/chat_data/fci_v2	Chatroom profile
/chat_data/mci_v2	Chatroom profile (miniaturized)
/chat_data/oci_v2	Open chatroom cover
/chat_data/tti	Temporary folder
/chat_data/url_image_v2	User profile Bot response #Search
/Contacts/fpi_v2	User profile
/Contacts/mpi_v2	User profile (miniaturized)
/DigitalItem/x2.0/etm_v2	Emoticon
/Moim/Media	Boards
/Moim/Media/thumbnail	Boards (miniaturized)
/OCH/img_v2	Open chatroom cover

Input: pragma, userId, encThumb
Output: decThumb

```
1: key ← pragma || userId
2: key ← MD5(key.getBytes())
3: iv ← MD5(Base64.encode(key))
4: return AES128/CBC/PKCS#7_DEC
        (encThumb, key, iv)
```

Fig. 12. Method for decrypting a thumbnail

카카오톡은 사용자 프로필 사진, 대화방 프로필 사진, 이모티콘 등 다양한 썸네일을 저장하지만 썸네일 중 중요한 분석 대상은 대화 중 전송된 미디어 파일에 대한 썸네일이다. 그러나 파일명을 암호화하여 저장하기 때문에 특정 메시지에 대한 썸네일을 식별할 수 없다. 이를 식별하기 위해서는 대화 기록 데이터베이스를 기반으로 썸네일을 분석할 필요가 있다. 미디어 파일이 전송되면 대화 기록 데이터베이스 내

에는 Fig. 13-15.와 같이 저장된다.

특히 attachement 컬럼 내에는 JSON 객체가 저장되며 미디어 파일을 획득할 수 있는 URL 경로와 token 값을 포함한다. URL을 통해 미디어 파일을 손쉽게 획득할 수 있지만 카카오톡 서버 내 저장 기간이 한정되어 있으므로 메시지가 전송된 후 일정 기간만 가능하다. token은 암호화된 썸네일 파일명을 생성하는 데 사용되는 입력 파라미터로 사진은 "k", 동영상은 "tk"를 key로 하여 저장된다. 암호화된 썸네일 파일명을 생성하는 방안은 다음과 같다. 하드 코딩된 바이트 배열에 대해 MD5 해싱을 수행하면 key가 된다. key에 대해 Base64 인코딩 수행 후 MD5 해싱을 수행하면 iv가 된다. 생성한 key, iv 쌍을 이용하여 token에 대해 AES 암호화 수행 후 SHA1 해싱을 수행하면 암호화된 썸네일 파일명이 생성된다. 이를 통해 특정 메시지에 대한 썸네일을 식별할 수 있으며, 구체적인 알고리즘은 Fig. 16.과 같다.

사용자 행위에 따른 썸네일의 변화를 분석하면 특정 썸네일의 존재 사실로 사용자의 각종 행위를 입증

	logId	message	attachement
1	2869490929879046144	사진	{"thumbnailUrl":"http:₩/₩/th-…
2	2869491111792281600	동영상	{"url":"http:₩/₩/dn-v.talk.kakao.com₩/talkv…

Fig. 13. Media type message


```
{
    "cs": "80193A02F8A2548230556671EAB367C92713EB7B",
    "h": 37,
    "k": "bJaDpi/oZnYbZe6js/KxUKziaOekNgHtvlVyAwGK/i_abcd55204664.png",
    "mt": "image/png",
    "s": 172,
    "thumbnailHeight": 93,
    "thumbnailUrl": "
http://th-m1.talk.kakao.com/th/talkm/oZnYbZe6js/KxUKziaOekNgHtvlVyAwGK/
i_abcd55204664_120x93.png",
    "thumbnailWidth": 120,
    "url": "
http://dn-m.talk.kakao.com/talkm/oZnYbZe6js/KxUKziaOekNgHtvlVyAwGK/i_ab
cd55204664.png",
    "w": 48
}
```


Fig. 14. Photo type message (attachement column)


```
{
    "cs": "901DD6CFAB6EF32860EFE783B81F2EA0598F5DE9",
    "d": 2,
    "h": 264,
    "s": 31458,
    "tk": "byHDyk/wrRVAp0Weg/vi4DtCoW6xnBtwQQdOUYmk/talkv_high.mp4",
    "url": "
http://dn-v.talk.kakao.com/talkv/wrRVAp0Weg/vi4DtCoW6xnBtwQQdOUYmk/
talkv_high.mp4",
    "w": 284
}
```


Fig. 15. Video type message (attachemer column)

Input: token
Output: encName

```
1: key ← hardcoded byte array
2: key ← MD5(key)
3: iv ← MD5(Base64.encode(key))
4: encName ← AES128/CBC/PKCS#7_ENC
           (token.getBytes(), key, iv)
5: encName ← SHA1(encName)
6: return toHexString(encName)
```

Fig. 16. Method for generating a thumbnail file name

할 수 있다. 최근 불법 성적 촬영물 및 아동·청소년성 착취물에 대한 소지, 시청 죄가 신설된 바 있다. 이러한 각종 행위를 입증할 수 있도록 대화방에서 일어나는 사용자 행위에 따른 썸네일의 변화를 전송, 열람, 삭제 행위 관점에서 분석하였다. 이때, "/chat_data/cli", "/chat_data/cli/thumbnail" 디렉토리에서 썸네일의 변화를 관찰할 수 있었으며, 이를 정리하면 Table 5.와 같다.

사용자가 미디어 파일을 발신하거나 수신한 미디어 파일을 기기 내에 저장하면 기기 내 미디어 파일의 저장 경로가 talkmedia.edb 데이터베이스 파일 tokenInfo 테이블 filePath 컬럼 내에 Fig. 17.과 같이 저장된다. 이때, talkmedia.edb 파일은 "%LocalAppData%\Kakao\KakaoTalk\users\{userDir}\chat_data" 디렉토리 내에 위치하며,

Table 5. Thumbnail changes by user behavior

Behavior		Change	Target directory
Transmit	Receive photo	△	thumbnail
	Receive video		
	Send photo	○	
	Send video		
	Forward photo	-	
	Forward video		
View	Open chatroom	△	thumbnail
	Scroll chatroom		
	Open photos/videos archives		
	Scroll photos/videos archives		

Table 5. Continued

Behavior		Change	Target directory
View	Open received photo	○	cli
	Open received video	-	
	Open sent photo		
	Open sent video		
	Delete sent photo from the device and open it in the chatroom	○	cli
	Delete sent video from the device and open it in the chatroom	-	
Delete	Delete for me	x	cli thumbnail
	Delete for everyone		
	Leave chatroom	-	

○: Thumbnail created
△: Create only thumbnails visible on the current chatroom/archives
x: Thumbnail deleted
-: No change

	token	url	filePath
1	bQpsvQ/oZn0dvO2h0/…	https://dn-m.talk.kakao.com/talkm/…	C:₩사진.png
2	dtUKr2/wrSdnjEtxq/…	https://dn-v.talk.kakao.com/talkv/…	C:₩동영상.mp4

Fig. 17. tokenInfo table

대화 기록 데이터베이스 파일과 같이 복호화 가능하다. 이를 통해 불법적인 미디어 파일의 소지, 유포 행위를 입증하고 사용자 기기 내 해당 미디어 파일이 저장된 경로를 식별할 수 있다.

IV. 결 론

메신저 사용자가 증가하면서 이를 범죄에 악용하는 경우가 많아지고 있으며, 이 외에도 각종 범죄 수사 시 메신저 사용 중 남겨진 대화 기록이 중요한 증거로 활용되기도 한다. 이러한 중요성에도 불구하고 각종 안티 포렌식 행위가 증가하며 메신저에 대한 디지털 포렌식 수사의 어려움은 점점 가중되고 있다.

본 논문에서는 국내에서 가장 많은 사용자를 가진 메신저인 카카오톡에 대해 Windows 환경에서 데이터 복호화 및 아티팩트 분석 방안에 관한 연구를 수행하였다. 카카오톡 사용 중 생성되는 각종 암호화된 데이터를 복호화하는데 필요한 입력 파라미터인 userId를 빠르게 획득할 수 있는 전수조사 방안, 메시지 삭제 시도 여부를 식별하고 모든 대화 상대에게서 삭제 시도한 메시지를 복호화하는 방안을 제시하고 썸네일 아티팩트 및 사용자 행위에 따른 썸네일의 변화를 분석하였다.

References

[1] Min-Dong Kim, Hyeon-Jin Lee, Sung Jin Lee, Yeon-Ju Lee, and Gi-Bum Kim, "A Study on the Permissibility of Search and Seizure of Mobile Messenger Data Using a USIM," Journal of The Korea Institute of Information Security & Cryptology, 31(2), pp. 197-209, Apr. 2021.

[2] Jaewon Choi and Seung-joo Kim, "A Study on Mobile Forensic Data Acquisition Method Based on Manufacturer's Backup Mobile App," Journal of The Korea Institute of Information Security & Cryptology, 28(1), pp. 95-110, Feb. 2018.

[3] Yeong-Woong Kim, Gi-Bum Kim, Ji-Hun Son, Yu-Ri Son, and Sung-Hyun Park, "Control Measures for Smartphone Forensics Using Fingerprint Information," Journal of Korean Public Police and Security Studies, 16(1), pp. 73-92, May. 2019.

[4] Jusop Choi, Jaegwan Yu, Sangwon Hyun, and Hyoungshick Kim, "Digital forensic analysis of encrypted database files in instant messaging applications on Windows operating systems: Case study with KakaoTalk, NateOn and QQ messenger," Digital Investigation, vol. 28, pp. S50-S59, Apr. 2019.

[5] Gyu-Won Lee, Seung-Jei Yang, Hyun-Uk Hwang, Kibom Kim, Taejoo Chang, and Ki-Wook Sohn, "A Recovery Scheme for the Deleted Overflow Data in SQLite Database," The Journal of Korean Institute of Information Technology, 10(11), pp. 143-153, Nov. 2012.
[6] Taejin Hwang, Dongho Won, and Youngsook Lee, "A study on the Comparison Analysis for Messenger Evidence Using Mobile Forensics," Convergence Security Journal, 18(2), pp. 25-32, Jun. 2018.
[7] Byungchan Jung, Jaehyeok Han, Hoyong Choi, and Sangjin Lee, "A Study on the Possibility of Recovering Deleted Data through Analysis of SQLite Journal in Messenger Application," Journal of Digital Forensics, 12(2), pp. 11-20, Sep. 2018.
[8] Gu-Min Kang and Seung-Kyu Kim, "A Study on the Mobile Forensic Method in the Case of Industrial Technology Leakage Using Smartphone," Korean Journal of Industrial Security, 10(2), pp. 7-35, Sep. 2020.
[9] Seung-Kyu Kim, Mu-Seok Kim, and Gu-Min Kang, "A Study on the Development of Mobile Forensic Tool for the Response to Hidden Camera Crime," Journal of Digital Forensics, 14(3), pp. 290-304, Sep. 2020.
[10] Ji Su Lee, Yeon Ju Lee, and Gi Bum Kim, "A Study on the Digital Forensic of Viewing Digital Sexual Exploitation Material in Mobile Messenger," The Journal of Police Science, 20(4), pp. 227-253, Dec. 2020.
[11] Seunghee Seo, Yeog Kim, and Changhoon Lee, "Instagram Users Behavior Analysis in a Digital Forensic Perspective," Journal of The Korea Institute of Information Security & Cryptology, 28(2), pp. 407-416, Apr. 2018.
[12] Daeho Yun and Sang Jin Lee, "Study for Android Smartphone's Gallery Thumbnail Forensic Analysis," KIPS Transactions on Computer and Communication Systems, 6(1), pp. 31-42, Jan. 2017.

〈저자소개〉

조 민 욱 (Minuook Jo) 정회원
2019년 2월: 명지대학교 수학과 이학사
2022년 8월: 세종사이버대학교 정보보호대학원 공학석사
〈관심분야〉 디지털 포렌식, 정보보호

장 남 수 (Nam Su Chang) 종신회원
2002년 2월: 서울 시립대학교 수학과 이학사
2004년 8월: 고려대학교 정보보호 대학원 공학석사
2010년 2월: 고려대학교 정보경영공학전문대학원 공학박사
2010년 2월~6월: 고려대학교 정보경영공학전문대학원 연구교수
2010년 7월~현재: 세종사이버대학교 정보보호학과 부교수
〈관심분야〉 암호칩 설계 기술, 부채널 공격, 공개키 암호 알고리즘, 공개키 암호 암호분석