Journal of The Korea Institute of Information Security & Cryptology ISSN 1598-3986(Print)
VOL.30, NO.3, Jun. 2020 ISSN 2288-2715(Online)
https://doi.org/10.13089/JKIISC.2020.30.3.405

WeChat 메신저의 향상된 복호화 방안과 SQLite Full Text Search 데이터를 이용한 삭제된 메시지 복구에 관한 연구*

허 욱,[1†] 박 명 서,[1] 김 종 성[2‡]
[1,2]국민대학교 (대학원생, 교수)

Study on Improved Decryption Method of WeChat Messenger and Deleted Message Recovery Using SQLite Full Text Search Data*

Uk Hur,[1†] Myungseo Park,[1] Jongsung Kim[2‡]
[1,2]Kookmin University (Graduate student, Professor)

요 약

스마트폰의 보급률이 늘어남에 따라 모바일 포렌식은 현대 디지털 포렌식 수사에서 필수적인 요소다. 모바일 메신저 데이터는 사용자의 생활패턴, 심리상태 등의 정보를 획득할 수 있기 때문에 모바일 포렌식에서 매우 중요한 데이터이다. 메신저 데이터 분석을 위해서는 암호화된 메신저 데이터의 복호화 기술이 필요하며, 대부분의 메신저가 메시지 삭제 기능을 제공하므로 삭제된 메시지를 복구하는 기술이 요구된다. 전 세계 약 10억 명이 사용하고 있는 메신저인 WeChat은 IMEI (International Mobile Equipment Identity) 정보를 이용하여 데이터를 암호화하며, 메시지 삭제 기능을 제공한다. 본 논문에서는 IMEI 정보가 존재하지 않는 경우의 데이터 복호화 방안을 제시하였으며, SQLite 데이터베이스의 전문 검색기능을 위하여 생성된 FTS (Full Text Search) 데이터베이스를 사용하여 삭제된 메시지를 복구하는 방법에 대하여 제안한다.

ABSTRACT

With the increase in smartphone user, mobile forensics has become an essential element in modern digital forensic investigation. Mobile messenger data is very important data in mobile forensics because it can acquire information such as user's life pattern and mental state. In order to analyze messenger data, a decryption technique of an encrypted messenger data is required. Since most messengers provide a message deleting function, a technique for recovering deleted messages is required. WeChat Messenger, a messenger used by about 1 billion people around the world, uses IMEI (International Mobile Equipment Identity) information to encrypt data and provides message deletion function. In this paper, we propose a data decryption method in the absence of IMEI information and propose a method for recovering deleted messages using FTS (Full Text Search) database created for full-text search function of SQLite database.

Keywords: Digital Forensics, Mobile forensics, Instant messenger, Data recovery

Received(03. 03. 2020), Modified(04. 23. 2020), Accepted(04. 23. 2020)

* 이 성과는 정부(과학기술정보통신부)의 재원으로 한국연구재단의 지원을 받아 수행된 연구임(No. NRF-2019R1F1A1060634)

† 주저자, gjdnr123@kookmin.ac.kr
‡ 교신저자, jskim@kookmin.ac.kr(Corresponding author)

I. 서 론

스마트폰 사용이 대중화되면서 모바일 인스턴트 메신저(이하 메신저)는 스마트폰을 활용한 통신의 필수적인 요소가 되었다. 메신저는 기존의 문자 메시지와 달리 인터넷을 사용하여 메시지를 전송하기 때문에 별도의 사용요금이 없으며, 대용량의 파일 공유가 가능하다. 또한, 일부 메신저의 경우 결제 및 계좌 송금 등의 금융기능을 포함한 다양한 기능을 제공하고 있다. 따라서 스마트폰 메신저 데이터를 분석하면 기본적인 대화 내용 외에도 사용자의 행적에 관한 다양한 정보를 획득할 수 있다. 그러나 대부분의 메신저는 사용자 데이터 보호를 위하여 데이터를 암호화하여 저장하고 있으며, 사용자가 선택적으로 특정 메시지를 삭제할 수 있는 기능을 제공하고 있다. 따라서 디지털 수사에서 메신저 데이터를 활용하기 위해서는 데이터 복호화 및 삭제된 메시지의 복원 기술이 필요하다. 또한, 스마트폰은 사생활 침해 우려가 있으므로 압수수색이 제한되며, 증거의 선별수집이 어렵다. PC 버전의 메신저 데이터는 동기화 기능을 통하여 모바일 메신저와 동일한 데이터를 포함하고 있으며, 비교적 간단하게 수집이 가능한 디지털 증거이다. 따라서 PC 버전의 메신저 데이터 또한 디지털 수사에서 유용하게 활용할 수 있다.

본 논문에서는 전 세계 약 10억 명 이상의 사용자를 가지고 있는 WeChat 메신저의 향상된 데이터 복호화 방안을 제시하며, SQLite 데이터베이스에서 제공하는 전문검색 기능의 색인 데이터를 활용하여 삭제된 메시지를 복구하는 방법에 대하여 서술한다. 1장에서는 배경지식으로 SQLite 데이터베이스의 암호화 모듈인 SQLCipher와 SQLite FTS 데이터베이스에 대하여 기술하였다. 2장에서는 관련 연구와 본 연구의 기여에 관하여 기술하였으며, 3장에서는 분석 환경과 데이터 추출방법을 기술하였다. 4장과 5장에서는 데이터 복호화 방법 및 실험을 통한 Android 및 Windows용 WeChat 메신저의 메시지 삭제 기능의 분석결과와 FTS 데이터베이스를 이용한 삭제된 메시지를 복원하는 방법에 대하여 기술하고, 6장에서 결론으로 마무리한다.

1.1 배경지식

본 절에서는 WeChat을 포함한 다양한 애플리케이션의 데이터베이스 암호화에 사용되는 SQLCipher의 데이터 암호화 방식에 관하여 설명한다. 또한, 본 연구의 삭제된 메시지 복구에 사용한 SQLite FTS 데이터베이스의 구조에 대하여 설명한다.

1.1.1 SQLCipher

SQLCipher는 오픈소스 형태로 제공되는 SQLite 데이터베이스의 암호화 모듈이다. SQLCipher는 SQLite 데이터베이스를 페이지 단위로 암호화하며, 암호화 알고리즘으로는 AES256-CBC를 사용한다. 암호화에 사용되는 키는 외부에서 입력된 PassPhrase와 공개된 매개변수(salt, 반복횟수)를 사용한 PBKDF2 알고리즘으로 생성한다. 여기서, PBKDF2의 PRF(Pseudo Random Function)는 HMAC-SHA1, HMAC-SHA256 등을 이용한다. 또한, 암호화된 데이터의 무결성을 검증하기 위하여 HMAC-SHA1, HMAC-SHA256, HMAC-SHA512를 선택적으로 사용할 수 있으며, 경우에 따라 HMAC을 이용한 무결성 검증 기능을 비활성화시킬 수 있다.

1.1.2 SQLite Full Text Search Extension

SQLite의 데이터베이스는 전문검색을 위하여 가상 테이블을 생성하는 확장 모듈을 제공한다. 최신 버전은 FTS5 이며, WeChat의 Windows버전은

Table 1. FTS database table structure

Table name		Contents
FTS3/4	FTS5	
〈name〉_content		Contains the actual data inserted into the FTS table. This shadow table is not present for contentless or external content FTS tables.
〈name〉_MetaData		Remaining data from the original table that is not contained in the 〈name〉_content table.
〈name〉_segdir	〈name〉_data	This table contains most of the full-text index data.
〈name〉_docsize		Contains the size of each column of each row in the virtual table in tokens. This shadow table is not present if the "columnsize" option is set to 0.

FTS4, Android 버전은 FTS5를 사용한다. 각 버전별로 생성되는 가상 테이블의 이름 및 내용은 Table 1.과 같으며, 〈name〉은 FTS 기능을 활성화한 원본 테이블의 이름이다[1][2].

〈name〉_segdir, 〈name〉_data 테이블에는 효율적인 검색을 위하여 원본 메시지를 토큰화하여 저장하고 있으며, 〈name〉_content 테이블에 원본 테이블의 데이터가 남아있다. 〈name〉_content 테이블에 포함되지 않은 원본 테이블의 데이터는 〈name〉_MetaData 테이블에 남아있다. 따라서 FTS 데이터베이스를 분석하여 원본 테이블의 내용을 확인할 수 있다.

II. 관련 연구 및 기여

2.1 관련 연구

WeChat의 암호화된 데이터베이스를 복호화하는 방법에 대한 다양한 연구가 존재한다. SongyangWu 등은 Android용 WeChat의 암호화된 데이터베이스의 복호화 방법과 SNS 기능의 데이터를 분석하였다[3]. 또한, SQLCipher의 PassPhrase 생성에 사용되는 UIN과 IMEI를 CompatibilityInfo.cfg 및 system_config_prefs.xml 파일로부터 획득할 수 있음을 보였다. WeChat의 경우 사용자가 중국에 집중되어 있기 때문에 국제 학술지 외에도 CSDN (Chinese Software Developer Network) 및 看雪安全과 같은 중국 개발자 커뮤니티에서 활발한 연구가 이루어지고 있다. 관련 연구 내용으로는 Windows 버전의 WeChat 데이터베이스의 복호화를 위하여 디버거를 활용하여 메모리에서 암호화에 사용된 PassPhrase를 읽어오는 방법 및 데이터 복호화 방법에 대한 분석결과가 존재한다[4]. 또한, 7.0 버전 미만에서 평문으로 존재하던 FTS 데이터베이스인 FTS5IndexMicroMsg.db 파일이 7.0 버전 이상에서부터 FTS5IndexMicroMsg_encrypt.db의 암호화된 데이터베이스로 변경되었으며, UIN, IMEI 및 WeChat ID 정보를 이용하여 복호화하는 방법에 대한 분석결과가 존재한다[5]. WeChat은 Android용과 Windows용 모두 SQLCipher를 사용하여 주요 데이터베이스를 암호화하며, 암호화에 사용되는 입력값은 Table 2.와 같다. UIN은 WeChat에서 규정한 각 사용자에게 부여되는 10자리 숫자로 된 고유번호이며, WeChat ID는 wxid로 시작되는 계정정보이다. 두 정보 모두 shared_prefs 폴더 내의 com.tencent.mm_preferences.xml 파일에서 획득할 수 있다. IMEI는 15자리 숫자로 된 단말기 고유 일련번호이며, 기기정보 및 다이얼에 *#06#을 눌러 확인할 수 있다. LEFT7은 16진 문자열의 앞 7자리 문자열을 의미한다.

SQLite 데이터베이스의 메시지 삭제는 delete와 secure delete 명령어를 사용하여 이루어진다. 일반적인 delete 명령어로 삭제된 경우 비할당 영역에 존재하는 데이터를 분석하여 삭제된 메시지를 복구할 수 있다[6]. secure delete 명령어는 delete 명령어와 달리 실제 데이터를 삭제하기 때문에 데이터베이스 파 일에서 메시지를 복구할 수 없다. secure delete를 사용한 경우 데이터베이스의 변경사항을 저장하는 저널링 데이터를 사용하여 데이터를 복구하는 연구가 진행되었다[7]. 본 논문에서 분석한 WeChat 메신저는 Android와 Windows 버전 모두 secure delete 명령어를 사용하여 메시지를 삭제하며, 최신 버전의 Windows용 WeChat은 프로그램 종료 시 저널링 데이터를 원본 데이터베이스에 적용한 뒤 삭제하기 때문에 저널링 데이터를 사용

Table 2. SQLCipher input values used for WeChat database encryption

OS	Target file	Page size	HMAC	KDF	Iteration	PassPhrase
Android	En*.db	1,024	X	PBKDF2-HMAC-SHA1	4,000	LEFT7 (MD5 (UIN || IMEI))
	FTS5 Index database	4,096	SHA1	PBKDF2-HMAC-SHA1	64,000	LEFT7 (MD5(UIN||IMEI||WeChat ID))
Windows	All database file	4,096	SHA1	PBKDF2-HMAC-SHA1	64,000	256bit Raw key

한 데이터 복원이 제한된다.

데이터베이스 분석을 통한 메시지 복구 방법 외에도 캐시 데이터 등의 데이터를 활용하여 삭제된 메시지를 복구하는 연구 또한 존재한다. 김기윤 등은 종단간 암호화를 지원하는 메신저인 SureSpot의 캐시 데이터를 분석하여 대화 내역에서 삭제된 미디어 파일을 복구하는 방법을 제시하였다[8].

2.2 기여

기존의 UIN과 IMEI를 사용한 복호화 연구결과는 2020년 1월 기준 최신 버전인 7.0.10 에서도 여전히 사용할 수 있다. 하지만 IMEI 정보의 접근이 거부된 기기 혹은 Wi-Fi 전용으로 출시된 태블릿과 같은 IMEI 정보가 존재하지 않는 기기의 데이터는 기존의 방법으로 복호화가 불가능하다. 본 논문에서는 소스코드 분석을 통하여 실제 암호화에 사용된 IMEI 정보를 암호화하여 저장하는 것을 식별하였으며, 이를 복호화하여 기존 방법으로 복호화가 불가능한 데이터를 복호화하는 방법을 개발하였다. 또한, 전문검색을 위한 색인 데이터인 FTS 데이터베이스를 분석하여 삭제된 메시지를 복구하는 방법에 대하여 제안하였다. WeChat은 Android용과 Windows용 모두 secure delete 옵션을 적용하였기 때문에 메시지 삭제가 일어나면 데이터가 완전히 지워진다. 저널링 파일을 활용한 데이터 복구의 경우 SQLCipher에 의하여 저널링 데이터 또한 암호화되어 있으며, 최근 삭제된 데이터에 대해서만 유효하다는 한계가 존재한다. 본 논문에서 데이터 복구를 위하여 제안한 FTS 색인 데이터는 메시지 삭제 여부와 관계없이 데이터가 남아있으며, 종료와 동시에 저널링 파일을 제거하는 Windows 버전에서도 활용할 수 있다.

III. 분석 환경 및 데이터 추출 방법

3.1 분석 환경

본 연구에서는 Android 및 Windows 용으로 출시된 WeChat에 대하여 분석을 진행하였다. 실험에 사용된 기기정보는 Table 3.과 같다.

애플리케이션 분석은 JAVA 코드 형태의 디컴파일을 지원하는 JEB decompiler[9]를 사용하여 분석을 진행하였다. 또한, 소스코드상으로 확인이 불가능한 부분은 IDA pro[10]를 사용한 동적분석을 통하여 실제 암호화에 사용된 값을 획득하였다. Windows용 프로그램은 Olly Dbg[11]를 사용한 디버깅을 통하여 메모리에 로드된 PassPhrase를 획득하였다. 암호화된 데이터베이스는 오픈소스로 제공되는 복호화 도구[12]를 사용하였으며, 해당 도구에서 제공되지 않는 공개된 복호화 방법은 일부 코드를 추가하였다. 또한, DB browser[13]를 통하여 데이터베이스의 내부 데이터를 확인하고, 비할당 영역은 HxD[14]를 사용하여 데이터 존재 유무를 확인하였다. RC4 복호화는 웹페이지 형태로 제공되는 JAVA script 기반의 오픈소스 복호화 도구인 cryptii[15]를 사용하였다.

Table 3. WeChat and OS version information of the target device

OS	Device	OS version	WeChat version
Android	Samsung Galaxy S10	10	7.0.9
	Xiaomi Redmi Note 5	9	7.0.10
	Amazon Fire HD 10 (2017)	6	7.0.9
Windows	-	Windows 10(1903)	2.7.1.88

3.2 데이터 추출 방법

Android 기기의 경우 OS의 보안정책으로 인하여 루트권한 획득 없이는 애플리케이션 내부 데이터의 접근이 불가능 하다. 구버전의 WeChat에서는 ADB (Android Debug Bridge) 백업[16]이 가능한 버전으로의 다운그레이드를 통해 데이터를 획득할 수 있었지만 최신 버전에서는 versionCode 옵션에 의하여 다운그레이드가 제한된다[17]. 본 논문에서는 ADB Shell 명령어를 통해 앱 데이터를 유지하면서 앱을 삭제한 뒤 ADB 백업이 가능한 버전으로 재설치하는 방식의 다운그레이드 방법을 이용하였다. 획득한 백업 데이터는 Android backup extractor[18]를 사용하여 내부 파일을 획득하였다. 다운그레이드를 위한 ADB Shell 명령어 및 추출된 데이터의 구조는 Fig. 1.과 같다.

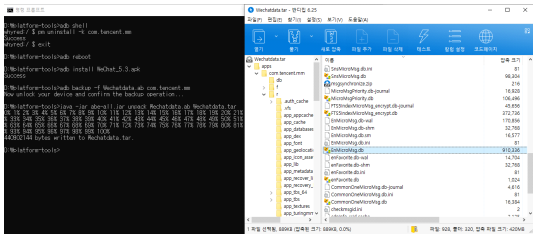

Fig. 1. ADB Shell command for extracting data and extracted WeChat data structures

Windows용 WeChat의 데이터는 〈user〉\Documents \WeChat Files\〈WeChat ID〉 경로에 존재하며, 특별한 권한없이 획득이 가능하다.

IV. WeChat 메신저의 데이터 복호화 방법

4.1 Android용 WeChat의 향상된 데이터 복호화 방법

기존 Android용 WeChat 데이터 복호화 방법은 키 생성에 필요한 IMEI를 기기 혹은 CompatibleInfo.cfg 파일 및 .auth_cache 데이터에서 가져오는 방법을 사용하였다. 하지만 일부 특수한 경우 IMEI가 남지 않는 것을 확인하였다. 첫 번째로 Android 버전이 6.0 이상인 경우 사용자가 애플리케이션의 권한을 항목별로 설정할 수 있도록 변경되었으며, Phone 권한을 허용하지 않은 경우 READ_ PHONE _STATE 권한이 없기 때문에 애플리케이션은 기기의 IMEI 데이터를 가져올 수 없다[19]. 두 번째로는 Wi-Fi 전용으로 출시된 태블릿의 경우 IMEI 정보가 존재하지 않는다. 마지막으로 최근 업데이트된 Android 10에서는 READ_PHONE_STATE 권한이 아닌 READ_PRIVILEGED_PHONE_STATE 권한이 요구되기 때문에 IMEI를 읽어올 수 없다[20]. IMEI를 획득할 수 없는 경우에도 WeChat 메신저는 데이터베이스를 암호화하기 때문에 기존 연구를 활용한 포렌식 도구는 IMEI 정보가 존재하지 않는 경우 데이터를 정상적으로 분석할 수 없는 것을 실험을 통하여 확인하였다. Fig. 2.는 실험에 사용된 스마트폰과 태블릿으로부터 추출한 데이터를 FINAL Mobile Forensics를 사용하여 분석한 결과이다.

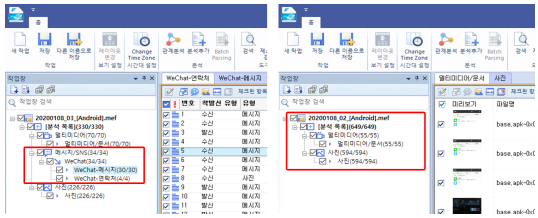

Fig. 2. Results of analysis using FINAL Mobile Forensics(Left : IMEI exist / Right : no IMEI)

이러한 한계점을 보완하기 위해 데이터베이스 암호화에 사용된 IMEI 관련 정보를 확인하기 위하여 애플리케이션 소스코드를 분석하였다. Fig. 3.은 IMEI 정보를 RC4 알고리즘을 사용하여 암호화하는 소스코드이다. 이때 RC4 암호화에 사용된 암호화 키는 소스코드상에 하드코딩되어 있는 "_wEcHAT_(0x5F7745634841545F)"을 사용하

Fig. 3. IMEI information encryption source code

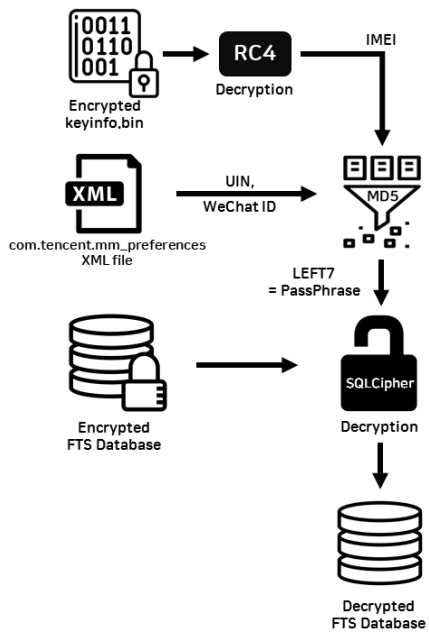

Fig. 4. Database decryption process using IMEI information obtained from keyinfo.bin file

며, IMEI 관련 정보는 암호화되어 com.tencent.mm/files/keyinfo.bin 파일에 저장된다.

추출된 파일을 복호화하여 확인한 결과, 일반적인 기기에서는 IMEI 값이 저장되어 있었으며, READ_PHONE _STATE 권한이 없거나 IMEI가 없는 태블릿 및 Android 10에서는 "1234567890ABCDEF"의 값을 IMEI 대신 사용하였다. Fig. 4.는 keyinfo.bin 파일로부터 획득한 IMEI 정보를 활용한 데이터베이스 복호화 과정이다. 결과적으로 keyinfo.bin 파일을 이용하면 앞서 설명한 세 가지를 포함한 모든 경우에서 IMEI 관련 정보를 획득하여 데이터 복호화가 가능하다.

4.2 Windows용 WeChat의 PassPhrase 획득 방법

Windows용 WeChat의 SQLCipher PassPhrase는 서버로 부터 받아오기 때문에 로그인된 모바일기기를 사용한 인증절차가 요구된다. 따라서 PassPhrase 획득을 위해서는 동일한 계정으로 로그인된 모바일 기기를 사용한 로그인 과정이 필요하다. 본 연구에서는 PassPhrase 획득을 위한 로그인 및 디버깅 과정에서 무결성을 유지하기 위하여 PassPhrase만을 획득하는 방법에 대하여 설명한다.

Windows용 WeChat은 아래 경로에 존재하는 .ini 파일을 서버로 전송하여 PassPhrase를 받아오기 때문에 다른 PC에서 로그인 하더라도 서버에 동일한 파일을 이용하면 동일한 PassPhrase를 획득할 수 있다.

o 〈user〉\Documents\WeChat Files\All Users\config

따라서 WeChat Files 폴더를 분석용 PC의 동일 경로에 복사한 뒤 디버거를 이용하여 분석하면 데이터베이스 암호화에 사용된 PassPhrase를 획득할 수 있다.

V. WeChat 메신저의 삭제된 메시지 복구

5.1 WeChat의 메시지 삭제 기능 분석

본 장에서는 Android 및 Windows용 WeChat 메신저의 메시지 삭제 기능에 대하여 설명한다. WeChat 메신저의 메시지는 아래 경로에 존재하는 데이터베이스 파일과 FTS 데이터베이스의 content 테이블에도 동시에 저장되며, Fig. 5.와 같이 사용자가 메시지를 선택적으로 삭제할 수 있는 기능을 제공한다.

o Android :
data/data/com.tencent.mm/MicroMsg/〈UIN MD5 hash〉/EnMicroMsg.db

data/data/com.tencent.mm/MicroMsg/〈UIN MD5 hash〉/FTS5IndexMicroMsg_encrypt.db

o Windows :
〈user〉\Documents\WeChat Files\〈WeChat ID〉\Msg\Multi\MSGn.db

〈user〉\Documents\WeChat Files\〈WeChat ID〉\Msg\Multi\FTSMSGn.db

메시지 삭제 기능을 이용하여 메시지를 삭제한 경우 데이터베이스에서 해당 메시지의 행이 제거되며, Fig. 6.과 같이 순차적으로 증가하는 메시지 id의 누락된 번호가 존재하기 때문에 메시지 삭제 여부를 식별할 수 있다.

메시지 삭제는 Android와 Windows 버전 모두 secure delete 옵션을 사용하기 때문에 Fig. 7.과 같이 데이터가 0으로 채워지면서 완전히 삭제되기 때문에 비할당 영역을 활용한 데이터 복구는 불가능하다.

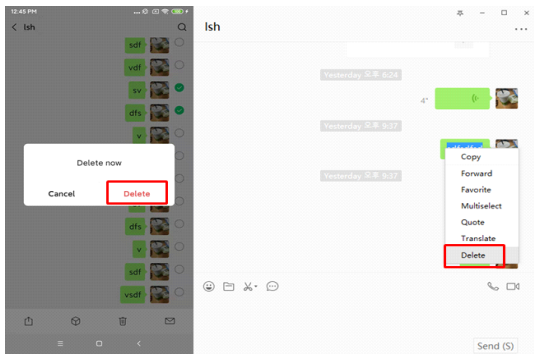

Fig. 5. Message delete function in WeChat Messenger (Left : Android / Right : Windows)

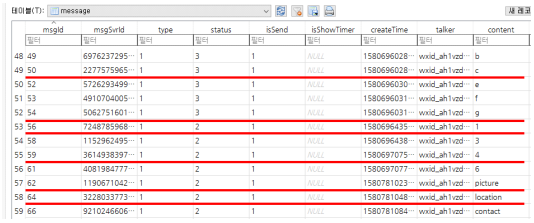

Fig. 6. Database missing message id due to message deletion

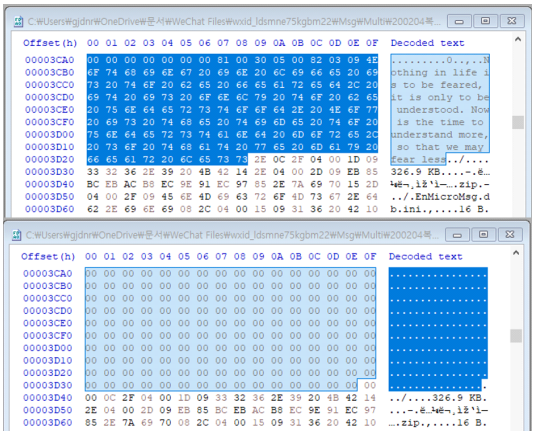

Fig. 7. Compare data before and after message deletion

secure delete 옵션을 사용하여 삭제된 메시지는 기존의 비할당 영역 분석 등을 통한 복구가 불가능하다. 삭제된 메시지가 미디어 파일 및 첨부 파일이면 아래 경로에 존재하는 파일 및 썸네일 이미지가 동시에 삭제된다. 〈user folder〉는 각 계정별로 생성되는 128bit 16진 문자열로 이루어진 폴더명이다.

모바일 (Android)

o 이미지

sdcard/tencent/MicroMsg/〈user folder〉/image2

o 동영상

sdcard/tencent/MicroMsg/〈user folder〉/video2

o 음성

sdcard/tencent/MicroMsg/〈user folder〉/voice2

o 파일

sdcard/tencent/MicroMsg/〈user folder〉/attachment

PC (Windows)

o 이미지

〈user〉\WeChat Files\〈WeChat id〉\FileStorage\Image

o 동영상

〈user〉\WeChat Files\〈WeChat id〉\FileStorage\Video

o 파일

〈user〉\WeChat Files\〈WeChat id〉\FileStorage\File

실험을 통하여 확인한 결과 카메라를 선택하여 즉시 촬영한 사진을 전송한 경우 메시지를 삭제하더라도 썸네일 이미지만 제거되는 것을 확인하였다. 또한, 첨부파일을 전송한 경우 EnMicroMsg.db의 appattatch 테이블에 파일 전송시각, 파일 크기 및 원본 파일 경로가 메시지 삭제 여부와 관계없이 남아있다.

Android와 Windows용 WeChat의 삭제 행위는 별개로 이루어지며, 서로 동기화되지 않는다. 실제로 Android에서는 삭제된 메시지가 Windows에는 남아있는 것을 확인하였다. Android에서 로그인된 PC의 여부를 확인하는 방법으로는 메시지 데이터베이스 파일이 존재하는 경로에 PC와 동기화 과정에서 생성되는 msgsyncronize.zip 파일의 존재 여부를 확인하는 방법과 대화상대 중 PC로의 파일 전송을 위하여 존재하는 filehelper의 존재 여부를 통하여 확인할 수 있다.

5.2 WeChat의 FTS 데이터베이스 분석

Android용 WeChat은 7.0 버전 이상부터 FTS 데이터베이스에 암호화를 적용하였다. 암호화는 동일하게 SQLCipher를 사용하며, Passphrase는 UIN, IMEI 및 WeChat ID를 연접하여 MD5 해시함수에 입력한 해시값의 앞 7자리 16진 문자열을 사용한다.

Android용 WeChat 7.0 버전 이전의 FTS 데이터베이스는 평문 상태로 존재하며, FTS5Index Message_content 테이블에 원본 메시지가 남아있다. 7.0 버전 이전의 FTS 데이터베이스는 메시지 삭제가 일어날 경우 secure delete 옵션이 적용하지 않는다. 따라서 누락된 메시지 id값을 확인하여 Fig. 8.과 같이 비할당 영역을 분석하여 메시지를 복구할 수 있다.

하지만, 모바일용 WeChat 7.0 버전 이후 및 PC 버전의 경우 FTS 데이터베이스에도 secure delete 옵션이 적용되어 있으며, Windows용 WeChat의 경우 최근 업데이트로 저널링 파일이 로

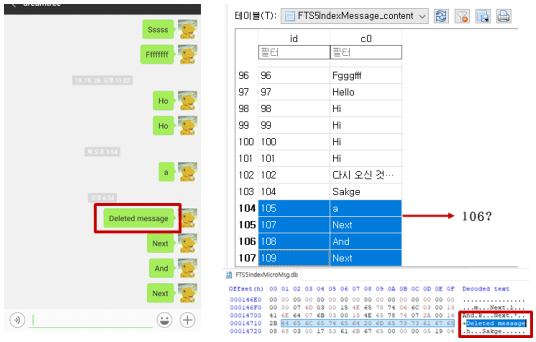

Fig. 8. Deleted messages that remain in unallocated space in older versions of FTS databases

그아웃과 동시에 적용되도록 변경되어 WAL 파일을 활용한 제한적인 복구 또한 불가능하다. 따라서 본 장에서는 FTS 데이터베이스의 색인 데이터를 활용하여 삭제된 메시지의 전체 혹은 일부를 복원하는 방법에 대하여 설명한다.

5.3 Android용 WeChat의 FTS 데이터베이스를 사용한 삭제된 메시지 복구

Android용 WeChat은 FTS5를 사용하기 때문에 〈name〉_data 테이블에 BLOB 데이터 형식으로 토큰화된 색인 데이터가 남아있다. 색인 데이터는 단일 메시지에 대하여 전체 메시지와 단어별 1~5글자의 데이터가 순차적으로 남아있다. Fig. 9.는 "test message"의 색인 데이터이며, 0 뒤로 전체 메시지가 남아있고 1~5의 숫자 뒤로 각 단어별 5글자까지의 데이터가 남아있는 것을 확인할 수 있다.

색인 데이터는 원본 메시지를 삭제하더라도 정상적으로 확인할 수 있다. 색인 데이터는 기본적으로 메시지가 생성된 순서로 남아 있으며, 일정한 주기로

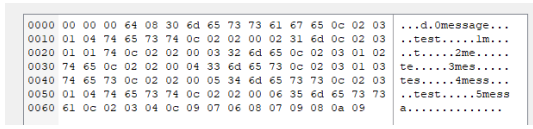

Fig. 9. FTS index data in WeChat for Android

병합된다. 색인 데이터가 병합된 경우 메시지의 id를 기반으로 삭제된 메시지의 전후 메시지를 이용하여 데이터를 추적할 수 있다. 실제 색인 데이터는 Fig. 10.과 같이 효율적인 검색을 위하여 단어 간의 순서는 알파벳순으로 0x3D('=')를 구분자로 사용하여 저장되며, 앞부분이 동일한 단어는 동일한 부분을 제거한 뒤 저장된다. 예를 들어 be, fear, in, less와 같이 알파벳 순으로 남아 있으며, that, the, time, to 의 경우 th와 t의 동일한 부분을 제외하고 that, e, ime, o 로 남아있다. 이때, 동일한 문자의 개수는 바이너리 데이터에 남아있는 것을 확인하였다. 따라서 여러 단어로 된 메시지는 알파벳 순으로 재배치 되기 때문에 재배치 작업이 필요하다. 결과적으로 순서는 변경되었지만 누락된 단어는 존재하지 않으므로 재배치 작업을 거치면 원본 메시지를 복구할 수 있다.

유니코드(UTF-8)로 이루어진 메시지의 경우 1개의 문자를 1~4 Byte로 표현하기 때문에 문자 단위로 색인 데이터가 생성된다. 문자의 순서는 영문과 유사하게 유니코드 순서로 0x41을 구분자로 사용하여 동일한 부분을 제거한 뒤 저장되며, 한글의 경우 가나다순과 일치한다. Table 4.는 "유니코드 메시지 테스트" 메시지에 대한 유니코드와 색인 데이터이다.

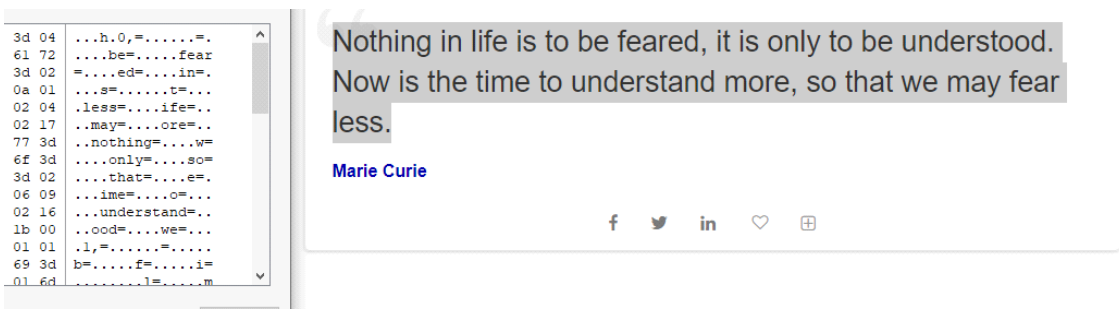

Fig. 10. Example of FTS index data

Table 4. Index data of unicode messages

Original message		Sorted message	
Korean	UTF-8	Korean	UTF-8
유	EC 9C A0	니	EB 8B 88
니	EB 8B 88	드	EB 93 9C
코	EC BD 94	메	EB A9 94
드	EB 93 9C	스	EC 8A A4
메	EB A9 94	시	EC 8B 9C
시	EC 8B 9C	유	EC 9C A0
지	EC A7 80	지	EC A7 80
테	ED 85 8C	코	EC BD 94
스	EC 8A A4	테	ED 85 8C
트	ED 8A B8	트	ED 8A B8
Indexed data			

```
00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F
00 00 00 97 04 30 EB 8B 88 41 02 03 02 02 93 9C
41 02 05 02 02 A9 94 41 02 06 01 03 EC 8A A4 41
02 0A 02 02 8B 9C 41 02 07 02 02 9C A0 41 02 02
02 02 A7 80 41 02 08 02 02 BD 94 41 02 04 01 03
ED 85 8C 41 02 09 02 02 8A B8 41 02 0B 00 04 31
EB 8B 88 41 02 03 02 02 93 9C 41 02 05 02 02 A9
94 41 02 06 01 03 EC 8A A4 41 02 0A 02 02 8B 9C
41 02 07 02 02 9C A0 41 02 02 02 02 A7 80 41 02
08 02 02 BD 94 41 02 04 01 03 ED 85 8C 41 02 09
02 02 8A B8 41 02 0B 04 08 07 07 08 07 07 07 07
08 07 09 07 07 08 07 07 07 07 08
```

5.4 Windows용 WeChat의 FTS 데이터베이스를 사용한 삭제된 메시지 복구

Windows용 WeChat은 Android와 달리 FTS4를 사용하기 때문에 〈name〉_segdir 테이블에 색인 데이터가 남아있다. Fig. 11.은 Windows

```
72 04  ..be.0.....fear.
73 05  0.....in.0....s.
6c 65  0......t.0....le
00 00  ss.0....ife.0...
30 15  .mai.0....ore.0.
03 30  ...noth.0....w.0
73 6f  ....onli.0....so
02 01  .0....that.0....
01 01  e.0....ime.0....
73 74  o.0......understo
0e 00  and.0....ood.0..
       ..we.0..
```

Fig. 11. FTS index data in WeChat for Windows

버전의 FTS 데이터베이스의 색인 데이터이며, Android와 달리 글자 수별 데이터는 별도로 저장하지 않은 것을 확인할 수 있다.

결과적으로 Android와 동일하게 색인 데이터를 통하여 원본 메시지를 복구할 수 있기 때문에 Windows 버전 또한 동일한 방법으로 삭제된 메시지의 복구가 가능하다.

Fig. 12.는 FTS 데이터베이스를 활용하여 삭제된 메시지를 복구하는 과정에 대하여 나타내었다.

VI. 결 론

모바일 메신저를 활용한 범죄 행위가 증가하면서 메신저 데이터는 현대 디지털 수사에서 주요 증거로 활용되고 있다. 하지만 대부분의 메신저에서 사용자 데이터 보호를 위해 제공하는 암호화 및 데이터 삭제 기능은 디지털 포렌식 관점에서 안티포렌식으로 작용한다.

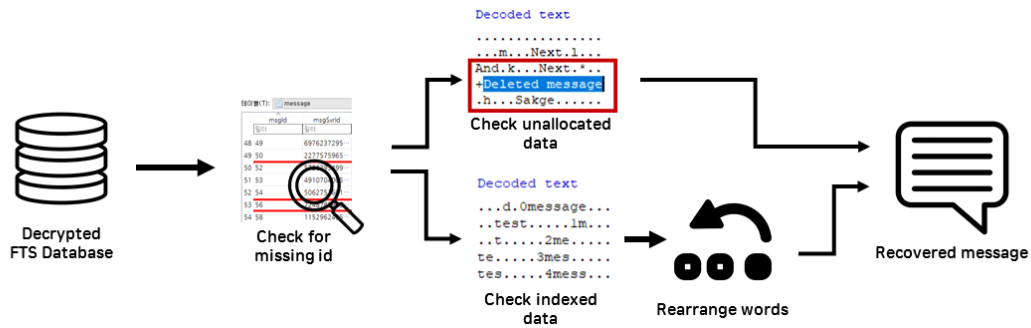

Fig. 12. Deleted message recovery process using FTS database

본 논문에서는 기존 방법을 적용할 수 없는 최신 OS 및 Wi-Fi 전용 태블릿의 경우에도 데이터를 추출 및 복호화할 수 있는 향상된 복호화 방안을 제시하였으며, 복호화된 데이터 분석을 통하여 삭제된 메시지의 복구 방안에 대하여 제안한다. 본 논문에서 제시한 방법을 사용하여 루팅되지 않은 Android 10이 설치된 스마트폰에서 성공적으로 데이터를 획득 및 복호화할 수 있었으며, 실험을 통해 복호화에 사용되는 기본값이 존재함을 확인하였다. 삭제된 메시지 복구는 SQLite에서 제공하는 전문검색 기능으로 인하여 생성된 FTS 데이터베이스의 색인된 데이터를 사용하였으며, 색인 데이터를 활용하여 삭제된 메시지를 복구하는 기술에 대하여 제안하였다. 전문검색 기능은 SQLite 외에도 MySQL 등 대부분의 데이터베이스에서 제공하고 있기 때문에 SQLite 뿐만 아니라 다른 DBMS (DataBase Management System)기반의 데이터베이스에도 활용될 수 있을 것으로 보인다.

References

〔1〕 SQLite, "FTS3 and FTS4 Extensions" https://www.sqlite.org/fts3.html

〔2〕 SQLite, "FTS5 Extension" https://www.sqlite.org/fts5.html

〔3〕 Wu, Songyang, et al. "Forensic analysis of WeChat on Android smartphones." Digital investigation, 21, pp.3-10, Jun. 2017

〔4〕 See Snow Security Forum, "Detailed tutorial for decrypting WeChat database in PC version" https://bbs.pediy.com/thread-251303.htm

〔5〕 See Snow Security Forum, "Android WeChat local database decryption and deleted chat history recovery complete tutorial" https://bbs.pediy.com/thread-250714.htm

〔6〕 SangJun Jeon, KeunDuck Byun, Jewan Bang, GuenGi Lee, SangJin Lee. "The Method of Recovery for Deleted Record in the Unallocated Space of SQLite Database." Journal of the Korea Institute of Information Security & Cryptology 21(3), pp.143-154, Jun 2011

〔7〕 Byungchan Jung, Jaehyeok Han, Hoyong Choi, Sangjin Lee. "A Study on the Possibility of Recover ing Deleted Data through Analysis of SQLite Journal in Messenger Application." Journal of Digital Forensics 12(2), pp.11-20, Sep 2018

〔8〕 Giyoon Kim, Uk Hur, Sehoon Lee, Jongsung Kim. "Forensic Analysis of the Secure Instant Messenger SureSpot." Journal of Digital Forensics 13(3), pp.175-188, Sep 2019

〔9〕 JEB Decompiler, "JEB Decompiler" https://www.pnfsoftware.com/

〔10〕 IDA Pro, "IDA Pro" https://www.hex-rays.com/products/ida/

〔11〕 OllyDbg, "OllyDbg" http://www.ollydbg.de/

〔12〕 Github, "Dump WeChat Messages from Android" https://github.com/ppwwyyxx/wechat-dump

〔13〕 DB Browser for SQLite, "DB Browser" https://sqlitebrowser.org/

〔14〕 HxD, "HxD" https://mh-nexus.de/en/hxd/

〔15〕 Cryptii, "Cryptii" https://cryptii.com/

〔16〕 Android Developers, "Android Debug Bridge" https://developer.android.com/studio/command-line/adb

〔17〕 Android Developers, "Set application version information" https://developer.android.com/studio/publish/versioning#appversioning

〔18〕 Github, "Android backup extractor" https://github.com/nelenkov/android-backup-extractor

〔19〕 Android Developers, "Manifest permission" https://developer.android.com/reference/android/Manifest.permission.html

〔20〕 Android Developers, "Privacy changes in Android 10" https://developer.android.com/about/versions/10/privacy/changes

〈저자소개〉

허 욱 (Uk Hur) 학생회원
2019년 2월: 건국대 신소재공학과 졸업
2019년 3월~현재: 국민대학교 금융정보보안학과 석사과정
〈관심분야〉 디지털 포렌식, 정보보호

박 명 서 (Myungseo Park) 학생회원
2013년 2월: 국민대학교 수학과 졸업
2015년 2월: 국민대학교 금융정보보안학과 석사
2014년 12월~2017년 2월: 국가보안연구소 연구원
2017년 3월~현재: 국민대학교 금융정보보안학과 박사과정
〈관심분야〉 디지털 포렌식, 암호 알고리즘, 정보보호

김 종 성 (Jongsung Kim) 종신회원
2000년 8월/2002년 8월: 고려대학교 수학 전공 학사/이학석사
2006년 11월: K.U.Leuven, ESAT/SCD-COSIC 정보보호 전공 공학박사
2007년 2월: 고려대학교 정보보호대학원 공학박사
2007년 3월~2009년 8월: 고려대학교 정보보호기술연구센터 연구교수
2009년 9월~2013년 2월: 경남대학교 e-비즈니스학과 조교수
2013년 3월~2017년 2월: 국민대학교 수학과 부교수
2014년 3월~현재: 국민대학교 일반대학원 금융정보보안학과 부교수
2017년 3월~현재: 국민대학교 정보보안암호수학과 부교수
〈관심분야〉 정보보호, 암호 알고리즘, 디지털 포렌식