안드로이드 모바일 환경에서의 인스턴트 메신저 아티팩트 분석

김성연,[1] 안예린,[1] 이서윤,[1] 김명주[2]

[1,2]서울여자대학교 (학부생, 교수)

Artifact Analysis of Instant Messengers in Android Mobile Environment

SungYeon Kim,[1] YeRin An,[1] SeoYun Lee,[1] MyuhngJoo Kim[2]

[1,2]Seoul Women's University(Undergraduate student, Professor)

요 약

법적 증거물로서의 모바일 포렌식의 중요성은 더욱 증가하고 있다. 본 논문에서는 국내외로 사용자가 많은 인스턴트 메신저인 Discord, Instagram, Band 어플리케이션의 아티팩트를 삭제 전과 후로 나누어 분석함으로써 디바이스 내에 저장되는 데이터와 그 데이터가 어느 경로로 저장되는지 파악하였고 주요 아티팩트를 정리하였다.

I. 서론

휴대폰, 태블릿 등 모바일 장치는 급속하게 성장하고 발전했다. 글로벌 여론조사기관인 '퓨 리서치(Pew Research)'에 따르면 대한민국의 휴대전화 보급률은 100%이고 스마트폰 보급률은 95%이다. 그에 따라 인스턴트 메신저라는 애플리케이션이 나오면서 텍스트, 음성, 사진 및 동영상 전송 등의 다양한 기능을 제공하기 때문에 현대인들이 필수적으로 사용하는 애플리케이션이 되었다. 이러한 메신저는 사용자의 행위에 따라 다양한 정보가 기록되며 해당 정보는 디지털포렌식 수사 시 중요한 증거로 활용될 수 있다. 모바일 장치는 개인이 항상 휴대하고 있고 개인의 정보가 실시간으로 기록된다는 점에서 법적 증거물로서의 모바일 포렌식의 중요성은 더욱 증가하고 있다. 본 논문에서 다루고 있는 모바일 포렌식은 모바일 장치에 저장된 메시지. 통화 내역, 인터넷 기록, 사진 등의 디지털 정보를 입수하고 분석한다.

1.1 관련 연구

모바일 포렌식에 관한 기존의 연구 결과는 다음과 같다.

'모바일 포렌식 증거능력 확보 방안 연구'에서는 모바일 기기에 대한 선별 압수에 제약사항이 있다는 점과, 기술의 적합성 검증 및 추후 발생될 문제점에 대해 분석하였으며, 해당 내용에 대한 대안 방안을 제시하였다[1].

'모바일 포렌식 증거 수집방안 연구: 제조사 백업 앱 기반 데이터 획득 기법' 에서는 백업용 모바일앱을 사용하기 어려운 스마트폰을 대상으로 무결성 훼손 없이 데이터를 획득하는 과정을 설명하였다[2].

또 '안드로이드 스마트폰 포렌식 분석방법에 관한 연구'에서는 안드로이드 스마트폰을 분석하는 방법론을 설명하였다[3].

Ⅱ. 본론

2.1 분석환경

① 환경 구축

해당 분석은 안드로이드 운영체제를 대상으로, 최상위 권한인 루트 권한을 얻기 위해 루팅 과정을 거친 후 진행되었다. 분석 PC의 환경은 Windows10

x64 환경이었으며, 분석 안드로이드 기기는 Galaxy A8(SM-A800S) 안드로이드 버전 7.0을 대상으로 진행되었다.

② 사용 Tools

ADB는 Android 기반 장치의 디버깅에 사용되는 프로그래밍 도구로, 해당 분석에서 루팅 및 추출 파일을 전송하는 용도로 사용되었다.

Netcat은 네트워크 연결에서 데이터를 읽고 쓰는 프로그램으로, ADB와 연계해 추출 파일을 전송하는 용도로 사용되었다.

DB4S(DB Browser for SQLite)는 SQLite와 호환되는 데이터베이스 파일을 편집하는 도구로, 어플리케이션 DB 파일을 열람하는 용도로 사용되었다.

2.2 인스턴스 메신저 별 데이터베이스 분석

① Instagram(인스타그램)

Instagram 데이터베이스에서는 총 6개의 의미 있는 테이블이 존재한다. 데이터베이스에 저장되는 데이터는 Table 1과 같다.

Table 1. 데이터베이스 내 전체 테이블 정보[4]

Table Name	Content
Android_Metadata	Country Used
Messages	Sever Item ID, Client Item ID, Timestamp, Message Type, Text etc
Mutations	Contain User ID, Mutation Type
Session	Session Type
Sqlite_Sequence	Amount Of Data Stored In Each Table
Threads	Contain Last Activity Time, User ID, Thread Info

(1) Messages 테이블

Messages 테이블에는 Thread ID가 저장된다. 해당 아이디로 Thread 테이블에 저장된 아이디를 확인할 수 있다. 송수신한 메시지 내용, 메시지를 전송한 사용자 아이디, Message 타입 등이 저장된다. Message Type은 Reel Share, Action_Log, Like, Text 등으로 분류되어 저장된다.

② Band(밴드)

93077060로 생성되는 데이터베이스 파일에는 8개의 테이블이 존재하며, 데이터베이스에 저장되는 데이터는 Table 2와 같다.

Table 2. 데이터베이스 내 전체 테이블 정보

Table Name	Content
Android_Metadata	Country Used
Category_Info	Chat Information (Last Chat Invitation Created At, New Message Count, Chat Invitation Count, Last Option Chat Create At)
Channel_User	Information About Channel Users
Chat_Channel	Participated Channel Information
Chat_Message	One-On-One Chat Messages And Corresponding Channel Information
Fail_Message	Message Failed To Send
Message_Reaction	Included Channel ID, Message Update Time, Count, And Code Type
User_Metadata	Data Key, Data Value

(1) Channel User 테이블

Channel User 테이블에서는 해당 핸드폰에서 밴드를 사용할 시점부터 참여하게 된 채팅의 Band ID와 사용자 식별 번호를 저장한다. 해당 사용자의 프로필 사진을 저장한 URL과 현재 사용자가 밴드 앱을 사용 중인지에 대한 상태 등을 알 수 있다.

(2) Chat_Channel 테이블

사용자가 현재 참여한 밴드의 수를 알 수 있으며, 해당 채널의 고유한 아이디 값과 채널 이름이 저장된다. 채널마다 채널에 참여하고 있는 사용자의 수가 함께 저장된다. 또한, 마지막으로 밴드 채팅 사용 시간, 메시지 등을 함께 저장하고 있다.

(3)Chat_Message 테이블

Chat_Message 테이블을 확인해 보면 채팅을 나눈 채널의 ID 값을 먼저 저장하고 있다. 해당 메시지를 보낸 사용자의 이름을 암호화하지 않고 평문으로 저장하고 있으며, 나눈 대화 메시지 또한 데이터베이스에서 평문으로 저장되고 있다. 읽은 횟수와 메신저 채팅에 참여하고 있는 사용자의 수 등을 같이 저장하고 있다. 메시지 전송 상태는 SEND_SUCCESS(전송 성공)와 같은 방법으로 표기하고 있다.

(4) Fail_Message 테이블

실패한 메시지와 사용자 이름, 상태, 채널 ID 등에 대한 정보를 확인할 수 있다. 해당 메시지의 타입이 같이 기록된다.

2.3. 인스턴스 메시지 별 Message 아티팩트 분석

인스턴스 메시지 별 영문으로 된 텍스트를 전송하였다. 첫 번째 상황은 인스턴스 메신저에 메시지 전송 후, 해당 메시지를 삭제하지 않고 인스턴스 메시지 별 데이터베이스를 추출한 상황이다.

① Instagram(인스타그램)

Instagram의 메신저 내용을 저장하는 데이터베이스는 Direct 데이터베이스이며, Instagram의 버전은 230.0.20.108이다.

Message 테이블에서 전송한 메시지를 확인할 수 있다.

	thread_id	recipient_ids	timestamp	message_type	text	message
	필터	필터	필터	필터	필터	필터
1	103009491282066057682	34417363827	1648356604229246	text	안녕하세요	{"status":"UPLOADED","item_type":"te···
2	103009491282066057682	34417363827	1648356619771431	text	Hello	{"status":"UPLOADED","item_type":"te···
3	103009491282066057682	34417363827	1648356622553550	text	Hi	{"status":"UPLOADED","item_type":"te···
4	103009491282066057682	34417363827	1648363485394385	media_share	NULL	{"status":"UPLOADED","item_type":"m···
5	103009491282066057682	34417363827	1648425495674436	reel_share	NULL	{"status":"UPLOADED","item_type":"re···
6	103009491281743611776	6850346583	1648388770375722	media_share	NULL	{"status":"UPLOADED","item_type":"m···
7	103009491282989776995	51715429782	1650199871351300	text	Hello world!	{"status":"UPLOADED","item_type":"te···
8	103009491282989776995	51715429782	1650199883851963	text	This is instagram message.	{"status":"UPLOADED","item_type":"te···
9	103009491282989776995	51715429782	1650199909545583	media	NULL	{"status":"UPLOADED","item_type":"m···
10	103009491282989776995	51715429782	1650199916769911	text	It is so cute dog.	{"status":"UPLOADED","item_type":"te···
11	103009491282989776995	51715429782	1650200016795591	text	Thank you so much! 😻	{"status":"UPLOADED","item_type":"te···
12	103009491282989776995	51715429782	1650200463963030	media	NULL	{"status":"UPLOADED","item_type":"m···
13	103009491282989776995	51715429782	1650200466952690	media	NULL	{"status":"UPLOADED","item_type":"m···
14	103009491282989776995	51715429782	1650200469574353	media	NULL	{"status":"UPLOADED","item_type":"m···

Fig. 1. 삭제 전, Instagram 메시지

Message 테이블의 칼럼은 다음과 같은데, 그중 Message_Type은 메시지의 형식을 의미하는 속성이다. 문자 메시지는 Text, 링크를 공유했을 때에는 Link, 사진 및 동영상은 Media, 공유된 게시글은 Media_Share, 메시지 좋아요를 눌렀을 때 알림은 Action_Log 등의 타입을 갖는다.

해당 기록은 계정 사용자가 보낸 메시지 기록뿐만 아니라 받은 메시지의 기록 또한 함께 저장한다.

② Discord(디스코드)

Discord에 'Forensics'라는 이름의 채팅방을 개설하였다. 기본적으로 TEXT CHENNELS는 '#General'과 '#Forensics'가 생성된다. 이 두 채팅방에 대한 기록을 분석하였다. Discord의 버전은 122.7-stable이다.

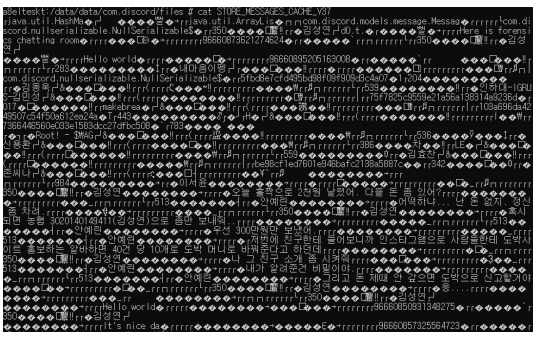


Fig. 2. Discord STORE_MESSAGES_CACHE_V37 파일

#Forensics 방에 보냈던 메시지는 메시지 캐시 파일에 저장되지 않았다. #General 방은 채팅방 별로 내용이 저장되는 게 아니며 모든 채팅방의 기록을 저장한다. 사용자가 보낸 메시지인 'Hello World' 등의 기록을 찾을 수 있으며, 이전에 타 사용자가 보냈던 메시지 또한 확인할 수 있다. 사용자가 보낸 메시지뿐만 아니라 타 사용자의 답장 메시지 기록까지 저장되며, 사용자의 이름과 함께 기록이 저장된다. 일정 범위만의 최신 메시지 기록을 저장하기에, 시간이 경과한 이전 메시지들의 기록을 찾을 수 없다.

Fig. 3. Discord STORE_MESSAGE_CACHE_V37 파일

'PBL5_친목방', 'forensic' 등 사용자가 참가한 채팅방의 기록을 찾을 수 있다. 해당 기록은 채팅방 명과 함께 채팅방에 가입한 시간을 함께 저장한다.

Fig. 4. Discord STORE_USERS_ME_V13 파일

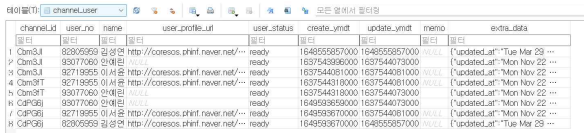
본 계정 사용자의 이메일과 이름을 담고 있다.

③ Band(Band)

Band Database 중, 93077060.db에서 추출한 내용이며, Band 버전은 7.10.0.3이다.

Fig. 5. Band chat_massage 테이블

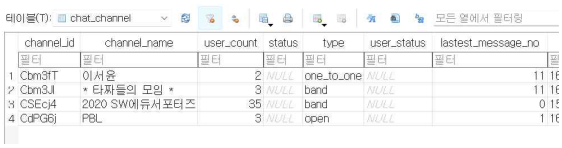
Chat_Message 테이블에서 메시지에 대한 정보를 찾을 수 있다. 채널의 고유한 ID, 이름, 멤버 수, 마지막으로 보낸 메시지 Number, 커버 이미지와 같은 채팅방의 정보를 볼 수 있으며, 고유한 id 값으로 메시지를 전송한 채널의 분류한다.



	channel_id	user_no	name	user_profile_url	user_status	create_ymdt	update_ymdt	memo	extra_data
	필터	필터	필터	필터	필터	필터	필터	필터	필터
1	Cbm3Jl	82806959	김성연	http://coresos.phinf.never.net/···	ready	1648555857000	1648555857000	NULL	{"updated_at":"Tue May 29 ···
2	Cbm3Jl	93077060	안예린	NULL	ready	1637543996000	1637544073000	NULL	{"updated_at":"Mon Nov 22 ···
3	Cbm3Jl	92719955	이서윤	http://coresos.phinf.never.net/···	ready	1637544081000	1637544081000	NULL	{"updated_at":"Mon Nov 22 ···
4	Cbm3fT	92719955	이서윤	http://coresos.phinf.never.net/···	ready	1637544318000	1637544081000	NULL	{"updated_at":"Mon Nov 22 ···
5	Cbm3fT	93077060	안예린	NULL	ready	1637544318000	1637544073000	NULL	{"updated_at":"Mon Nov 22 ···
6	CdPG6j	93077060	안예린	NULL	ready	1649593659000	1637544073000	NULL	{"updated_at":"Mon Nov 22 ···
7	CdPG6j	92719955	이서윤	http://coresos.phinf.never.net/···	ready	1649593670000	1637544081000	NULL	{"updated_at":"Mon Nov 22 ···
8	CdPG6j	82806959	김성연	http://coresos.phinf.never.net/···	ready	1649593670000	1648555857000	NULL	{"updated_at":"Tue May 29 ···

Fig. 6. Band channel_user 테이블

Channel_User 테이블을 통해서 Cbm3fT라는 채널에는 일대일 메시지로 전송하였다고 유추할 수 있다. CdPG6j 채널에는 세 명의 사용자와 1:N 채팅방으로 메시지를 전송했다고 유추할 수 있다.



	channel_id	channel_name	user_count	status	type	user_status	lastest_message_no
	필터	필터	필터	필터	필터	필터	필터
1	Cbm3fT	이서윤	2	NULL	one_to_one	NULL	11
2	Cbm3Jl	* 타짜들의 모임 *	3	NULL	band	NULL	11
3	CSEcj4	2020 SW에듀서포터즈	35	NULL	band	NULL	0
4	CdPG6j	PBL	3	NULL	open	NULL	1

Fig. 7. Band chat_channel 테이블

다음 테이블인 Chat_Channel을 보면 Cbm3fT 채널에서는 타입이 One To One이라고 표기되는 것을

볼 수 있다. 이로 인해 사용자는 타 사용자에게 일대일 메시지로 전송했다는 것을 확인할 수 있다.

CdPG6j 채널에서는 사용자가 채널을 생성했다는 정보와 함께 3명의 참가자들이 있는 것으로 보아 단체 채팅방에서 해당 내용을 전송했다는 사실을 알 수 있다. 최근 열람하거나 최근에 보낸 메시지만 저장을 한다. 즉, 보지 않거나 오래된 메시지는 해당 DB 파일에서 찾을 수 없다.

두 번째 상황은 인스턴스 메신저에 메시지 전송 후, 해당 메시지를 삭제한 후, 인스턴스 메시지 별 데이터베이스를 추출한 상황이다.

① Discord(디스코드)

Fig. 8. Discord의 메시지 기록

Discord의 경우, #General 메시지 방과 #forensics 메시지방에 작성했던 내용을 삭제한 뒤, 진행하였다. STORE_MESSAGES_CACHE_V37 파일은 메시지 캐시를 저장하는 파일이므로, 메시지 삭제 여부를 해당 파일을 통해 알아보았다. #General 톡 방에 있던 'Hello World' 등의 기록은 삭제 후에도 여전히 STORE_MESSAGES_CACHE_V37 파일에 남아있었지만, #forensics 방의 메시지는 찾을 수 없었다.

② Band(밴드)

Band Database 중, 위와 동일한 93077060.db에서 추출한 내용이다.

	channel_id	channel_name	user_count	status	type	user_status	lastest_message_no	create_ymdt
	필터	필터	필터	필터	필터	필터	필터	필터
1	Cbm3fT	이서윤	2	NULL	one_to_one	NULL	0	1637544318000
2	Cbm3JI	* 타짜들의 모임 *	3	NULL	band	NULL	11	1637543996000
3	CSEcj4	2020 SW에듀서포터즈	35	NULL	band	NULL	0	1582002851000
4	CdPG6j	PBL	3	NULL	open	NULL	0	1649593659000

Fig. 9. Band의 Chat_Channel 테이블

Chat_Channel 테이블을 보면 Channel_User 테이블과 같이 메시지를 삭제하기 전과 비교하였을 때 Cbm3fT 채널과 CbPG6j 채널을 제외하고, 변화가 없는 것을 확인할 수 있다. 이는 새롭게 채널이 생성되거나 멤버 수의 변화가 없다는 것을 알 수 있다.

channel_id	tid	message_no	type	message	ext_message	user_no	name
Cbm3fT	627392241	11	1			93077060	안예린
CdPG6j	627392732	1	1			93077060	안예린

Fig. 10. Cbm3ft 채널의 칼럼 값

Cbm3ft 채널은 위에 Lastest_Message_No 칼럼의 값이 11 이었다. CbPG6j 채널의 Lastest_Massage_No 칼럼의 값은 11 이었다. 해당 칼럼은 마지막으로 보낸 메시지의 개수를 의미한다.

status	local_ext_message
DELETED	NULL
DELETED	NULL

Fig. 11. Chat_Message 테이블

Chat_Message 테이블을 보면 위와 비교하였을 때 같은 id 값을 가진 메시지가 삭제된 것을 볼 수 있다. 삭제되기 전에는 SEND_ACCESS였던 Status 값이 DELETED로 변한 것을 볼 수 있다. 메시지를 삭제한 상태에서 본 결과, 내용을 더 이상 알 수 없게 되었지만 메시지를 전송했다는 사실과 메시지 삭제 행위를 한 것을 볼 수 있다.

Ⅲ. 결론

현대인이 사용하는 인스턴트 메시지에는 다양한 개인 정보들이 기록된다. 따라서 가장 빈번하게 사용되는 메신저들의 각 메신저가 저장하고 있는 데이터의 범위, 내용, 특징 등을 파악하고 메신저를 삭제하고 난 이후의 내용도 파악할 필요가 있다. 본 논문에서는 국내외로 사용자가 많은 인스턴트 메신저인 Discord, Instagram, Band을 대상으로 분석을 진행하였고, 각 메신저에 대한 아티팩트 수집 및 분석을 시행하였다. 각 애플리케이션의 아티팩트를 분석함으로써 디바이스 내에 저장되는 데이터와 그 데이터가 어느 경로로 저장되는지 파악하였고 주요 아티팩트를 정리하였다. 본 논문에서 분석한 각 어플리케이션들의 아티팩트 위치와 내용을 기반으로 인스턴트 메신저가 디지털 포렌식 업무 시 더욱 빠른 분석 및 증거 수집을 가능하게 하여 효율적으로 활용될 것을 기대한다.

[참고문헌]

[1] Soowoong Eo, Wooyeon Jo, Seokjun Lee, Taeshik Shon, Ensuring the Admissibility of Mobile Forensic Evidence in Digital Investigation, Feb. 2016

[2] Jaewon Choi, Seung-joo Kim, A Study on Mobile Forensic Data Acquisition Method Based on Manufacturer's Backup Mobile App, Feb. 2018

[3] Jung Hoon Oh, A Study for Android Smartphone Forensic Analysis, Dec. 2011

[4] Sumin Shin, Eunhu Park, Soram Kim, Jongsung Kim, Artifacts Analysis of Slack and Discord Messenger in Digital Forensic, April.2020