JDCS
디지털콘텐츠학회논문지
Journal of Digital Contents Society
Vol. 26, No. 5, pp. 1363-1373, May. 2025

Check for updates

iOS 환경에서 카카오톡 데이터 복호화 및 아티팩트 분석 연구

김 도 현[1] · 김 병 욱[2] · 양 영 욱[2*] · 장 홍 준[3*]
[1]전주대학교 컴퓨터공학과 학사
[2]국립강릉원주대학교 컴퓨터공학과 조교수
[3]강원대학교 데이터사이언스학과 조교수

Decryption and Artifact Analysis of KakaoTalk Data in the iOS Environment

Do-Hyeon Kim[1] · Byoungwook Kim[2] · Yeongwook Yang[2*] · Hong-Jun Jang[3*]
[1]Bachelor's Degree in Computer Engineering, Jeonju University, Jeonju 55069, Korea
[2]Assistant Professor, Dept. of Computer Science and Engineering, Gangneung-Wonju National University, Wonju 26403, Korea
[3]Assistant Professor, Dept. of Data Science, Kangwon National University, Chuncheon 24341, Korea

[요　　약]

메신저 대화 데이터는 사용자 기기에 암호화된 형태로 저장되며, 디지털 포렌식에서 중요한 증거로 활용된다. 특히 카카오톡 대화를 분석하려면 데이터베이스 파일을 추출한 후 복호화해야 한다. 기존 연구에 따르면 Windows 및 macOS용 카카오톡 데이터베이스는 AES 및 PBKDF2와 같은 암호화 기법으로 보호되며 복호화 방법이 연구되었지만, iOS용 데이터베이스 복호화에 대한 공개적 연구는 부족한 상황이다. 본 논문에서는 iOS용 카카오톡 데이터베이스를 복호화하는 방법을 제시하며, 데이터 백업 과정과 복호화 알고리즘을 설명한다. 또한 알고리즘의 정확도 검증을 위해 테스트 케이스를 설계하고 실험을 진행한다. 본 연구 결과는 아이폰을 이용한 범죄 수사에서 사용자 행위 및 사건 전개 분석에 기여할 것으로 기대된다.

[Abstract]

Messenger conversation datum are stored in encrypted form on user devices and are used as important evidence in digital forensics. In particular, to analyze KakaoTalk conversations, database files must be extracted and decrypted. According to previous studies, KakaoTalk databases for Windows and macOS are protected by encryption techniques, such as AES and PBKDF2. Although decryption methods have been studied, public research on decryption of databases for iOS is insufficient. This paper presents a method of decrypting KakaoTalk databases for iOS and explains the data backup process and decryption algorithm. In addition, a test case was designed, and the accuracy of the algorithm was experimentally verified. The results of this study are expected to contribute to the analysis of user behavior and case development in criminal investigations using iPhones.

색인어 : 카카오톡, 복호화, iOS 기기, 디지털포렌식, 데이터베이스
Keyword : KakaoTalk, Decryption, iOS Device, Digital Forensics, Database

http://dx.doi.org/10.9728/dcs.2025.26.5.1363

Received 06 April 2025; **Revised** 29 April 2025
Accepted 08 May 2025

***Corresponding Author; Yeongwook Yang, Hong-Jun Jang**
Tel: +82-33-760-8671, +82-33-250-8452
E-mail: yeongwook.yang@gwnu.ac.kr, hongjunjang@kangwon.ac.kr

Ⅰ. 서 론

메신저의 대화 내용이 디지털 포렌식에서 사용되고 있다. 관련 연구들에서는 카카오톡, WhatsApp, Slack 등의 메시지 복호화가 이루어졌으며, 이러한 연구는 데이터 복구에서 중요한 역할을 하고 있다.

각종 메신저의 대화 내용은 기기의 내부 저장소에 데이터베이스 형태로 암호화되어 저장된다. 데이터베이스에는 메시지 내용, 전송 시간, 발신자 및 수신자 정보, 멀티미디어 파일 경로 등 다양한 아티팩트가 포함된다.

그러나 은폐 행위(의도적인 조작, 은닉, 삭제 등)를 통해 이러한 대화 내용이 조작되거나 삭제될 수 있다. 따라서 디지털 포렌식을 위해 이러한 은폐 행위를 탐지하고 데이터의 무결성을 보장하는 기술이 중요하다.

상용 디지털 포렌식 소프트웨어(Magnet 사의 Axiom Magnet Foreniscs Axiom, 지엠디소프트 사의 MD-RED, BelkaSoft 사의 포렌식 도구 등)들은 메신저 데이터 복호화를 지원하지만, 복호화가 제대로 이루어지지 않는 경우가 종종 발생한다.

공개된 기술로 연구된 복호화 기술로는 Windows용 카카오톡 데이터베이스 복호화 및 macOS용 카카오톡 데이터베이스 복호화 연구가 있다. 그러나 iOS 기기의 카카오톡 데이터베이스 복호화에 대한 연구는 아직 진행된 바 없다.

본 연구는 iOS 환경에서 카카오톡 데이터베이스의 복호화 및 삭제된 메시지 복구를 직접 구현하고 실험적으로 검증한 최초 연구로, 기존 연구와의 차별성을 갖는다. 특히, iOS 백업 데이터를 활용한 데이터베이스 복호화 절차를 규명하고, HMAC(hash-based message authentication code) 및 XOR(exclusive OR) 연산을 활용한 파생키 생성 방식을 분석하여 iOS 환경의 복호화 로직을 제안하였으며, 삭제된 메시지 복구 알고리즘을 구현하고 실험을 통해 성능을 검증하였다.

연구 진행 과정은 먼저 iOS 기기에서 카카오톡 데이터를 추출하고, 이를 복호화하여 다양한 아티팩트를 식별 및 분석하는 단계로 구성된다. 그런 다음, 식별된 아티팩트를 기반으로 삭제된 메시지를 복원하는 알고리즘을 개발 및 테스트한다. 삭제된 메시지 복원 방법은 데이터베이스에서 삭제 로그와 잔존 데이터를 분석하여 복원 가능한 메시지를 추출하는 방식으로 이루어진다.

여기서 아티팩트(Artifact)란 사용자의 행위에 의해 생성된 다양한 데이터 조각을 의미하며, 예를 들어 메시지 내용, 전송 시간, 발신자 및 수신자 정보, 첨부 파일 경로, 삭제 로그 등이 포함된다. 이러한 아티팩트는 디지털 포렌식 수사에서 중요한 증거로 활용될 수 있다. 이 연구를 통해 디지털 포렌식 및 범죄 수사 분야에서 카카오톡 데이터를 더욱 효율적으로 활용하며, 동시에 개인정보 보호를 강화하는 데 기여할 것으로 기대된다.

본 연구의 기여는 다음과 같이 요약할 수 있다.

1) 복호화된 데이터의 아티팩트에 대한 상세한 분석

본 연구는 복호화된 카카오톡 데이터에서 추출된 아티팩트에 대해 상세한 분석을 수행하여, 메시지 내용, 전송 시간, 발신자 및 수신자 정보, 첨부 파일 경로, 삭제 로그 등 다양한 데이터 조각을 체계적으로 정리하고 이를 디지털 포렌식 수사에 활용할 수 있도록 한다.

2) 공개된 iOS 기기 카카오톡 데이터 복호화 방법 제안

본 연구는 iOS 기기에서 카카오톡 데이터를 효과적으로 복호화 할 수 있는 새로운 방법을 최초로 제안하며, 이 방법을 통해 암호화된 데이터베이스의 내용을 해독하고 중요한 증거를 확보하는 절차를 구체적으로 제시한다.

기존 연구들은 Android 및 Windows 기반 복호화 기법에 집중되어 있으며, iOS 플랫폼에 대해 복호화 방식 자체를 최초로 실험적으로 규명하고, 삭제 메시지까지 복구한 사례는 본 연구가 처음이다.

본 논문의 2장에서는 과거 연구된 카카오톡을 포함한 다른 메신저들의 복호화 연구를 다루며, 3장에서는 iOS 기기의 카카오톡 데이터베이스 복호화 방법, 4장에서는 실험 결과를 다룬다.

Ⅱ. 관련연구

2-1 메신저 데이터 복호화 관련 연구

기존의 메신저 데이터 복호화 연구는 사용자 관련 정보 및 대화 내역이 저장된 데이터베이스의 복호화와 관련 아티팩트에 대한 연구가 진행되었다.

1) PC 기반 복호화 연구

윈도우 환경에서 카카오톡 메신저의 백업 프로세스를 분석하여, 암호화된 데이터베이스의 암호화 절차와 암호키 생성 과정을 규명하였다 [1].

또한, 안드로이드 환경과 윈도우 환경에서 카카오톡 아티팩트를 비교 분석하는 연구도 진행되었으며, 이를 통해 각 환경에서만 남는 아티팩트를 특정하고 윈도우 환경에서의 메시지 유형별 행위를 명시하였으며, 안드로이드 상의 카카오톡 데이터와 아티팩트 유형을 분석하고, 백업파일을 통해 삭제된 메시지를 복구하는 연구도 진행되었다[2].

[3]에서는 WhatsApp과 데이터베이스 암호화 상태를 비교 분석하였으며, 각 애플리케이션에서 삭제된 메시지를 시각적으로 분석하는 방법을 기술하였다.

[4]에서는 윈도우 환경에서 사용되는 메신저의 AES (advanced encryption standard) 키 생성 및 데이터베이스 암호화 방식, 패스프레이즈 생성 과정을 연구하였고, 삭제된 메시지를 식별하고 복구하는 방법을 제시하였다. 또한, Slack

과 Discord의 아티팩트를 분석하여 각 메신저의 데이터가 어떻게 활용될 수 있는지 가상 시나리오를 제시하는 연구도 이루어졌다.

2) 모바일 기반 복호화 연구

이외에도 모바일 환경에서의 메신저 데이터 획득 방안을 제시하고, 획득한 데이터의 구조와 암호화 프로세스에 대한 연구가 진행되었으며, 로그인 패스워드 및 삭제된 메시지 복구 방안에 대해서도 연구하였으며, 삭제된 메시지를 복구하는 다양한 방법을 제시했다. 또한, SQLite 데이터베이스를 암호화하기 위한 확장 모듈인 SQLCipher를 이용한 암호화 기능을 제공하는 메신저의 암호화 과정을 분석하고 복호화 후 주요 아티팩트를 선별하여 정리하는 연구도 진행되었다[5].

안드로이드 환경에서는 SQLite의 저널링 방식을 이용한 메신저 삭제된 데이터 복구 가능성에 대한 연구가 이루어졌으며, 이 연구는 메신저별 설정되어 있는 저널 모드에 따라 삭제된 메시지 복구 방법이 다르다는 것을 언급하며, 메신저별 저널 모드를 확인하고 메시지 삭제 방법에 따른 복구 가능 여부를 제시했다[6].

또한, 스마트폰에 저장된 암호화된 소셜 네트워킹 서비스 애플리케이션 데이터(카카오톡 채널 관리자 등)를 식별하고 복호화하는 연구도 진행되었고[7], 카카오톡 애플리케이션이 아닌, Signal 및 Telegram 보안 메신저 디지털 포렌식 분석 연구도 진행되었다. 각 메신저의 주요 아티팩트와 데이터베이스 구조를 분석하여 포렌식 수사에 유용한 정보를 제공한다[8]. 추출된 데이터를 바탕으로 자연어처리 기술을 활용하여 메신저 데이터를 분석하고, 이를 통해 범죄 수사에 필요한 증거를 효과적으로 추출하는 연구도 진행되었다[9].

여러 논문에서 WhatsApp 포렌식 도구들의 기능과 분석 특징을 비교하는 연구[10] 및 안드로이드 스마트폰에서 WhatsApp 메신저의 포렌식 분석을 다루는 연구도 진행되었다[11]. WhatsApp뿐만 아니라, WeChat의 포렌식 분석을 다루며, 사용자 데이터의 추출 및 분석 방법을 제시한 연구도 진행되었다[12].

최근에는 macOS 환경에서의 카카오톡 데이터베이스 복호화를 위한 키 생성 및 복호화 방법에 관한 연구가 진행되었다. 해당 연구에서는 macOS에서 추출한 백업 파일 내부의 특정 plist 파일과 키체인 접근을 통해 AES 복호화 키를 획득하고, 이 키를 이용해 PBKDF2 방식으로 보호된 SQLite 데이터베이스 파일을 복호화하였다. 또한, 복호화 후 데이터베이스 구조를 분석하여 대화 내역과 사용자 정보가 저장된 테이블을 식별하고, 이를 통해 아티팩트를 정리하였다[13].

기존 연구들은 주로 Windows 및 macOS 환경에서 카카오톡 데이터베이스의 복호화 절차와 알고리즘 구조, 특히 AES와 PBKDF2를 중심으로 전체 데이터베이스 복호화를 수행한 바 있다. 그러나 iOS 환경에서의 카카오톡 데이터 복호화 및 삭제된 메시지 복구를 모두 실험적으로 구현하고 검증한 연구는 보고된 바 없다.

본 연구는 iOS 백업 데이터로부터 카카오톡 데이터베이스를 복호화하고, 삭제된 메시지를 복구하는 전체 과정을 구현하였다. iOS의 백업 구조, 암호화된 데이터베이스 파일, 복호화 키 생성 과정, SQLCipher 기반 복호화 절차, 그리고 삭제된 메시지 복구 방법까지 포괄적으로 다루었다. 본 연구와 기존 연구 간 주요 차이점은 표 1과 같다.

표 1. 선행 연구
Table 1. Previous research

Study	Target Platform	Encryption Method	Database Structure
[1] Minuook Jo et al. (2023)	Windows	AES, PBKDF2	EDB
[2] Nabi Lee (2022)	Android, Windows	AES-128	SQLite
[3] Byungchan Jung et al. (2018)	Android	SQLChipher	SQLite
Proposed method (2025)	iOS	AES-CBC, HMAC	SQLite

2-2 Windows와 iOS용 카카오톡의 차이점

Windows용 카카오톡 데이터베이스와 iOS 기기의 카카오톡 데이터베이스는 구조적으로 차이가 존재한다. Windows 환경에서는 각 채팅방을 개별 데이터베이스 파일로 저장하며, 연락처 및 관련 데이터도 별도의 데이터베이스로 관리된다. 따라서 데이터베이스마다 복호화 알고리즘이 다르게 적용된다. 반면, iOS 환경에서는 대화 기록과 연락처를 각각 하나의 데이터베이스로 저장하며, 두 데이터베이스에서 동일한 복호화 알고리즘이 사용된다. Windows와 iOS 환경의 차이점은 표 2와 같다.

표 2. Windows와 iOS의 차이점
Table 2. Windows & iOS difference

	Windows	iOS
Data storage method	Multiple individual DB files used	All messages stored in a single SQLite DB file
Encryption method	EDB (Entire database encryption)	Encryption at the individual record level
Decryption process	Entire EDB file is decrypted first then individual data is analyzed	Decryption based on a specific field (userid)

Windows 환경에서는 데이터베이스가 개별 채팅방별로 저장되므로 전체 파일을 해독한 후 데이터를 분석해야 한다. 반면, iOS 환경에서는 SQLite DB 파일 내에서 특정 레코드 단위로 암호화된 메시지를 개별적으로 복호화하는 방식이 적용된다. 또한, Windows 환경에서는 복호화 키 생성 과정이 복잡하지만, iOS 환경에서는 UserId 기반의 파생 키를 활용하여 상대적으로 단순한 방식으로 복호화가 이루어진다.

Ⅲ. 복호화 방안

본 연구에서 제안하는 복호화 방안은 iOS 기기의 전체 데이터 백업을 통한 데이터 추출, 카카오톡 데이터베이스 분석 및 주요 데이터 식별, 암호화된 메시지 복호화, 그리고 삭제된 메시지 복구의 절차로 구성된다.

3-1 iOS 기기의 데이터 추출

iOS 환경에서 카카오톡 데이터 획득은 Jailbreak 기법 또는 전체 데이터 백업을 통해서만 가능하다. Jailbreak은 iOS 및 iPadOS에서 내부적으로 접근하지 못하게 임의로 제한을 해제하는 방식이다. Jailbreak로 데이터를 획득하면 디지털 포렌식 무결성의 원칙에 위반된다.

따라서 디지털 포렌식의 증거 능력을 보존하기 위해서는 iOS 기기의 전체 데이터 백업을 진행해야 한다. 전체 데이터 백업은 해당 기기의 잠금이 해제되어야 가능하다. 만약 사용자가 잠금 해제에 동의하지 않으면 데이터 백업을 진행할 수 없다. 생체정보를 이용한 포렌식 방안은 아직 법제화되지 않아 강제로 잠금을 해제하는 것은 불가능하다. 따라서 해당 논문에서는 기기의 잠금이 해제되어 있다고 가정하고 진행한다.

iOS 기기의 전체 데이터 백업은 일반적으로 iTunes 프로그램을 사용하여 진행된다. iTunes를 사용하지 않고 백업하는 방법은 표 7에서 설명한다. iTunes를 사용한 백업 방법은 iOS 기기를 Windows 또는 macOS 환경에 연결한 후 iTunes 프로그램을 실행하여 백업을 진행하는 것이다. 이 과정에서 iOS 기기에 암호를 입력해야 하고, 암호를 입력하면 표 3의 위치에 전체 데이터 백업 파일이 생성된다. 데이터 저장 위치는 사용자가 임의로 변경하지 않는 한 동일 경로이다.

표 3. 데이터 백업 경로
Table 3. Data backup path

OS	Path
Windows	/Users/(username)/AppData/Roaming/Apple Computer/MobileSync/Backup/
macOS	~/Library/Application Support/MobileSync/Backup/

표 7은 iTunes를 사용하지 않고 명령줄 인터페이스(Command Line Interface; CLI)로 데이터를 추출하는 방식을 나타낸다. 표 4는 해당 CLI 도구의 위치를 보여준다.

표 4. Cli 데이터 백업 도구 경로
Table 4. Cli data backup tool path

OS	Path
Windows	C:/Program Files (x86)/Common Files/Apple/Mobile Device Support/AppleMobileBackup.exe
macOS	/System/Library/PrivateFrameworks/MobileDevice.framework/Versions/Current/AppleMobileDeviceHelper.app/Contents/Resources/AppleMobileBackup

AppleMobileBackup의 경우 iTunes와의 통신이 불필요로 하다. AppleMobileBackup을 사용하면 iTunes 프로그램을 실행하거나 iTunes를 통해 인증을 받지 않아도 iOS 기기의 데이터를 백업할 수 있다. 이는 분석 환경에서 불필요한 소프트웨어를 설치하거나 추가적인 인증 절차를 거치지 않고도, 필요한 리소스만 사용하여 데이터를 효율적으로 추출할 수 있다는 장점을 제공한다. AppleMobileBackup 프로그램에 사용되는 인자 값은 표 5와 표 6과 같다.

표 5. Applemobilebackup 프로그램 인자
Table 5. Applemobilebackup program arguments

Argument	Description
–b	Perform a backup
–r	Perform a restore
–e	Erase target device

표 5에서 ‘-b’는 데이터 백업을 위한 인자 값이다. 데이터 백업 위치는 표 3의 각 운영체제에 맞는 위치에, 디바이스의 UUID (universally unique identifier) 값을 폴더 이름으로 하여 저장된다. ‘-r’은 백업 데이터를 iOS 기기로 복원할 때 사용한다. ‘-e’는 데이터 백업 파일을 삭제하는 인자이다.

표 6. Applemobilebackup 프로그램 선택 인자
Table 6. Applemobilebackup program optional arguments

Argument	Description
–t deviceid	Target device identifier
–o path	Output file for action (not loggin)

표 6은 백업 데이터의 위치를 변경하거나 특정 기기 선택에 사용되는 옵션 파라미터이다. ‘-o’는 백업 데이터의 저장 경로를 변경하는 옵션이다. ‘-t’는 디바이스를 지정해 백업을 진행하기 위해서 사용된다. ‘-t’에 대한 하위 인자 값은 UUID 값이다. 인자 값 사용 예는 표 7과 같다. 본 연구에서는 표 7과 같은 CLI 데이터 추출 방법으로 iOS 기기의 전체 데이터를 백업한다.

표 7. Cli 데이터 백업 명령어
Table 7. Cli data backup command line

OS	Path
Windows	C:/Program Files (x86)/Common Files/Apple/Mobile Device Support/AppleMobileBackup.exe –b –target [UUID] –o [Destination]
macOS	/System/Library/PrivateFrameworks/MobileDevice.framework/Versions/Current/AppleMobileDeviceHelper.app/Contents/Resources/AppleMobileBackup –b –target [UUID] –o [Destination]

표 7에 필요한 인자는 표 5, 6에서 확인할 수 있다. CLI에서 전체 데이터를 백업하는 경우 UUID 값이 필요하다. UUID는 iOS 기기와 물리적으로 연결되어야 확인할 수 있다.

UUID를 획득하는 방법은 리눅스 및 유닉스 계열 운영체제에서 USB 장치의 정보를 표시하는 명령어 lsusb를 사용하며, 예시는 표 8과 같다.

표 8. UUID 획득 방법
Table 8. Obtaining UUID

Detail
lsusb -v 2> /dev/null | grep -e "Apple Inc" -A 2

획득한 UUID를 표 8에 적용하면 iOS 기기의 전체 백업을 진행할 수 있다. 그림 1에서는 전체적인 카카오톡 데이터 획득 과정을 나타내며 복호화 절차는 그림 2와 같이 정리된다.

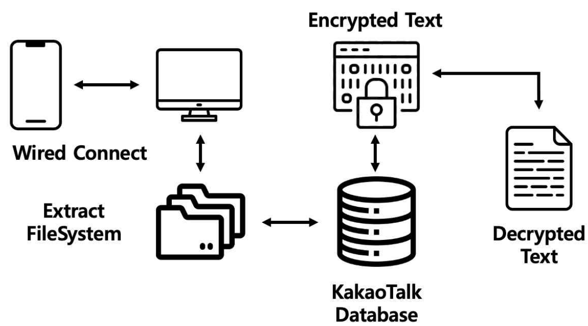


그림 1. 카카오톡 데이터 획득 전체 과정
Fig. 1. Overall KakaoTalk data acquisition process

본 논문에서는 전체 데이터 백업을 통하여 데이터에 접근하며, 암·복호화 과정을 알고리즘으로 구현한다. 분석 대상은 표 9와 같으며, 분석 도구로는 표 10의 도구를 사용한다.

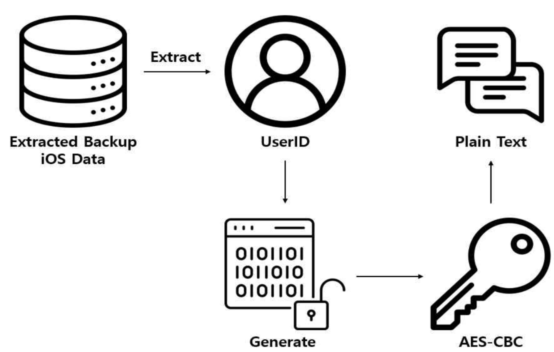


그림 2. iOS 기반 카카오톡 메시지 복호화 절차도
Fig. 2. Decryption procedure of KakaoTalk messages on iOS

표 9. 분석 대상
Table 9. Analysis target

Classification	Detail
Model	iphone 15 pro
Os	ios 17.4.1
Kakaotalk	10.7.1

표 10. 분석 도구
Table 10. Analysis tools

Name	Version
Python	3.11
DB browser for sqlite	3.12.2

3-2 추출된 데이터 구성

iOS용 카카오톡 데이터의 저장 경로는 표 11과 같다. 대화 내역 등 주요 정보는 모두 데이터베이스에 저장되어 있으며, 주고받은 멀티미디어 파일은 사용자별 디렉터리에 저장된다. 이는 그림 3과 같이 모두 암호화된 상태로 저장되어 있다.

message	userId
필터	필터
Sav4N6jmycJEhJ0lSNcee9dN/...	274970908
ZE+h+BssDlW6ecnCUc58/g==	155834097
T8rhlL5bRmp/iz32f0Glx4oOx6ZQrpecFkEBONkj73g=	155834097
rxJ58KxAlBrDAaWhuD8DkQ==	155834097

그림 3. Message.sqlite 내 암호화된 메시지
Fig. 3. Encrypted messages within message.sqlite

표 11. 카카오톡 데이터 경로
Table 11. Kakaotalk data path

Name	Path
Message database	/AppDomain-com.iwilab.KakaoTalk/Library/PrivateDocuments/Message.sqlite
Talk database	/AppDomain-com.iwilab.KakaoTalk/Library/PrivateDocuments/Talk.sqlite
Multimedia file	/AppDomain-com.iwilab.KakaoTalk/Library/Library/PrivateDocuments/chat/[ChatID]/[Image FileName]

3-3 암호화된 메시지 복호화 방안

본 연구에서는 iOS 환경에서 카카오톡 데이터베이스의 복호화 로직을 직접 규명하였다. iOS 백업 데이터를 역분석하여 SQLite 기반의 DB 구조를 분석하였으며, userId + HMAC + XOR 연산을 통한 파생키 생성 과정을 도출하였다. 특히, 기존 연구에서는 iOS 환경의 카카오톡 복호화 방식이 알려지지 않았으나, 본 연구에서는 실제 iOS 백업 데이터를 활용하여 이를 규명하고, 복호화 알고리즘을 구현하였다.

기존 연구들은 Windows 및 macOS 환경에서 EDB 또는 SQLite 데이터베이스 자체를 복호화하는 방식을 사용하였으나, iOS 환경에서는 각 메시지 레코드가 개별적으로 암호화되며, 복호화를 위해 userId를 기반으로 한 HMAC 해시 연산과 XOR을 활용한 파생키 생성 방식이 필요하다.

복호화에 필요한 사용자 userId는 Message 테이블 내 userId 칼럼에서 확인할 수 있다. 메시지 복호화는 HMAC로

해시를 계산하여 수행된다. HMAC은 해시 함수와 비밀 키를 사용하여 메시지의 무결성과 인증을 보장하는 기법이다. 주로 SHA-256과 같은 해시 함수를 이용하며, 두 번의 해싱 과정을 통해 보안을 강화한다. 주어진 메시지에 대해 두 번의 HMAC를 계산하고 두 결과를 XOR 연산하여 최종 해시를 생성한다. 이후 UserID에 16개 NULL 문자를 추가하여 총 16바이트로 패딩한다. 최종 해시를 이어 붙여 32바이트의 파생키를 생성한다. 생성된 파생키로 AES 알고리즘을 초기화하고, CBC (cipher block chaining) 모드로 설정한다.

AES는 128비트 블록 크기와 128, 192, 256비트 키 길이를 사용하며, 여러 라운드의 바이트 치환, 행 이동, 열 혼합, 라운드 키 더하기 과정을 통해 데이터를 암호화한다. CBC는 블록 암호화 모드 중 하나로, 각 블록이 이전 암호화된 블록과 XOR 연산 후 암호화되는 방식이다. 초기 벡터가 첫 번째 블록과 XOR 연산에 사용되며, 이를 통해 동일한 평문이라도 다른 암호문이 생성된다. 입력 메시지를 Base64로 디코딩한 후 AES로 복호화한다. 복호화된 결과를 디코딩하고, 패딩을 제거하면 암호화된 메시지가 복호화된다. 구체적인 알고리즘은 표 12와 같다.

표 12. 암호화된 메시지 복호화 방법
Table 12. Method for decryption a encrypted message

```
Input : key, iv, userId, encMessage
Output : decMessage
1 : userId ← userId + &₩0& * 16
2 : key ← hardcoded byte array
3 : iv ← hardcoded byte array
4 : First ← HMAC(key, userId + hardcode)
5 : Second ← HMAC(key, userId + hardcode)
6 : DerivedKey ← XOR(First + Second)[:32]
7 : encoder ← AES(DerivedKey, AES.MODE_CBC, iv)
8 : decrypted ← Base64.decode(encMessage)
9 : decrypted ← decrypted[:-decryptd[-1]]
```

표 12의 입력 파라미터인 key, iv는 각종 데이터를 암호화하는 데 흔히 사용된다. 하드 코딩된 바이트 배열을 key와 iv로 하여 XOR 연산을 통하여 AES 복호화를 수행한다. 위 결과에 대해 Base64 디코딩을 수행하면 복호화 된 메시지를 확인할 수 있다. 입력 파라미터 중 userId는 카카오톡 회원가입 시 사용자별로 부여되는 시퀀스로 기기 내에서 획득할 수 있다. 표 12의 구체적인 알고리즘 설명은 다음과 같다.

1행에서는 주어진 사용자 ID에 '₩0' 문자를 16개 추가하여 사용자 ID를 16배수로 맞춘다. 이것은 AES 암호화 알고리즘의 특성이며, 블록 크기에 맞추기 위한 것이다.

2, 3행에서는 키 및 초기화 벡터 할당을 진행하며, 키와 초기화 벡터는 코드에 하드코딩된 바이트 배열이다. 4, 5행에서는 First 및 Second 계산한다. HMAC 함수를 사용하여 주어진 키와 사용자 ID에 하드코딩된 값을 추가한 것에 대한 해시값을 계산하고, 이 과정을 두 번 수행하여 First와 Second 값을 얻는다. 6, 7행에서는 DerivedKey를 계산한다. First와 Second 값을 XOR 연산한 후, 결과를 32바이트로 제한하여 키를 얻는다. 또한, AES 암호화에 사용될 키와 초기화 벡터를 설정한다.

8행에서는 주어진 암호화된 메시지를 Base64 디코딩 하여 암호화된 이진 데이터를 얻으며, 9행에서는 AES의 패딩 방식에 따라 마지막 바이트에 패딩 크기가 저장되어 있으므로, 이를 사용하여 패딩을 제거한다. 위 알고리즘을 사용하면 카카오톡 메시지 복호화를 위한 복호화 키를 얻을 수 있다. 표 12의 알고리즘을 수행하면 그림 4와 같이 복호화된 데이터를 확보할 수 있다.

332458112785663	120828260	1	679286416	test message 1
332458112785663	222013892	1	679286728	test message 2
332458112785663	120828260	1	679286762	test message 3

그림 4. 복호화된 메시지
Fig. 4. Decoded message

표 11의 Talk Database 내 ZUSER 테이블은 표 13에 제시된 정보를 포함하고 있으며, ZACCOUNTID, ZCHATID, ZID, ZNICKNAME 등 주요 데이터를 저장하고 있다. 해당 데이터베이스의 주요 칼럼 정보는 표 13과 같다.

표 13. Talk.sqlite 테이블 설명
Table 13. Description for Talk.sqlite

Column	Description
ZCHATID	Chat id information of the associated chat
ZID	Unique user id for each individual
ZCUSTOMNAME	Custom name set by the owner
ZNAME	Custom name set by the owner
ZNICKNAME	Nickname set by the friend
ZPHONENUMBER	(encrypted message) friend's phone number

표 11의 Talk Database에서 소유주 UserId를 확인하는 알고리즘은 표 14와 같다.

표 14. iOS 소유자 userid 찾기 방법
Table 14. Method for finding iOS owner userid

```
Input : Talk.sqlite
Output : UserId, UserNickName
1 : db ← open(Talk.sqlite)
2 : db ← EXECUTE(&SELECT ZID,
    ZNICKNAME, COUNT (*) AS Count FROM
    ZUSER GROUP BY ZID ORDER BY Count
    DESC LIMIT 1&)
3 : db ← FETCHALL()
4 : UserId ← db[0][0]
5 : UserNickName ← db[0][1]
```

자체 제작한 도구로 본 논문에서 언급한 복호화 알고리즘을 사용하면 그림 5, 6과 같이, 채팅방 목록, 채팅 상세 내용들을 모두 복호화할 수 있다. 그림 5는 실제 iOS 기기에서 전체 데이터 백업은 진행, 추출, 복호화가 완료된 모습이다.

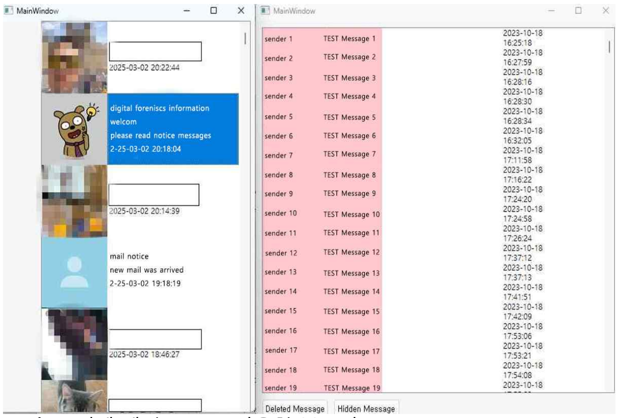


그림 5. 자체 제작 도구를 이용한 복호화
왼쪽) 채팅방 목록, 오른쪽) 채팅 내용
Fig. 5. Decoded with a self-created tool
left) list of chat rooms, right) chat contents

그림 6. 복호화된 채팅 내용 일부
Fig. 6. Partially decoded chat contents

Windows와 macOS 환경에서는 데이터베이스가 EDB 형식으로 암호화되어 데이터베이스 접근에 시간이 소요되며, UserID를 알아내는 데에도 시간이 소요된다. 하지만 iOS 기기의 경우 데이터베이스를 암호화하는 방식이 아닌 Message.db 파일 내 레코드들의 데이터만 암호화하는 방식이기 때문에 비교적 쉽게 UserID에 접근할 수 있다.

3-4 삭제된 메시지 복호화 방안

1) 모든 대화 상대에게서 삭제 복호화 원리

그림 7과 같이 모든 대화 상대에게서 삭제 기능으로 메시지를 삭제하더라도, 해당 메시지는 데이터베이스에서 삭제되는 것이 아니다. 카카오톡의 Message 데이터베이스의 Message 테이블에는 serverLogId, type, clientMsgId, message, attachment 등의 필드가 있다.

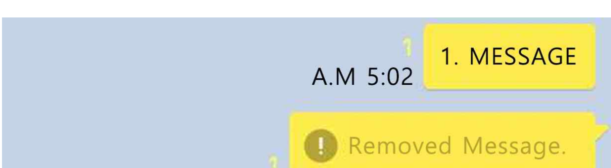


그림 7. 모든 대화 상대에게서 삭제
Fig. 7. Delete from all interlocutors

모든 대화 상대에게서 삭제 기능으로 메시지를 삭제하면, type 필드 값이 16386으로 변경되며, 그림 8과 같이 로그가 추가된다(표 15 참조).

표 15. 메시지 삭제 후 데이터베이스 변경
Table 15. Change the database after deleting the message

{"feedType":14,"logId":[logId],"hidden":true,"byHost":false}

type	clientMsgId	readAt	message
16386	Filter	Filter	Filter
16386	0	665751549	HWGjTXLUFXlezdk5USMpdg==
16386	0	665751550	HWGjTXLUFXlezdk5USMpdg==
16386	0	665751550	HWGjTXLUFXlezdk5USMpdg==
16386	0	718382859	HWGjTXLUFXlezdk5USMpdg==

그림 8. 모든 대화 상대에게서 삭제된 메시지
Fig. 8. Deleted message from all interlocutors

해당 로그는 logId에 맞는 내용을 자동으로 숨김 처리하지만, 실제로는 데이터베이스에 여전히 남아 있어 디지털 포렌식에서 중요한 단서가 될 수 있다.

모든 대화 상대에게서 삭제된 메시지는 여전히 데이터베이스에 존재하므로, 이를 복구가 가능하다. 하지만 '이 기기에서만 삭제' 기능을 사용하여 메시지를 삭제할 경우, message 레코드가 비워져 삭제된 메시지를 복구할 수 없다(그림 9 참조). 사진, 동영상 등의 멀티미디어 파일은 thumnailurl을 통해 파일에 접근할 수 있다.

type	clientMsgId	readAt	message
Filter	Filter	Filter	Filter
12	0	697604644	
20	0	697607144	

그림 9. 이 기기에서만 삭제 후 물리적 데이터베이스 변경
Fig. 9. Change the physical database after deleting only on this device

2) 보충 설명 및 삭제된 메시지 복구 알고리즘

삭제된 메시지 복구를 위해서는 데이터베이스에서 type 필드와 logId 필드를 분석하여 삭제된 메시지를 식별하고, 이를 복호화하는 과정이 필요하다. 이를 위해 다음과 같은 절차를 따른다.

(1) 데이터베이스 연결 및 쿼리 실행

Talk.sqlite 파일을 열고, Message 테이블에서 type 필드 값이 16386인 레코드를 찾은 후에, 해당 레코드의 logId 값을 추출한다.

(2) 숨겨진 메시지 복구

logId 값을 기반으로 삭제된 메시지를 포함한 레코드를 찾아 복호화한다. 또한 logId에 대응하는 원본 메시지를 message 필드에서 복구한다.

(3) 멀티미디어 파일 복구

삭제된 메시지 중 멀티미디어 파일의 경우, thumnailurl 필드를 통해 파일을 확인하고 복원한다.

이 절차를 알고리즘으로 나타내면 표 16과 같다. 이 절차

를 통해 모든 대화 상대에게서 삭제된 메시지와 멀티미디어 파일을 복구할 수 있다.

표 16. 삭제된 메시지 복구 방법
Table 16. Method for deleted message

```
Input : Message.sqlite, key, iv
Output : DeleteMessage

db ← open(Message.sqlite)
db ← EXECUTE(&SELECT userId, message
      FROM Message WHERE type=16386&)
rows ← FETCHALL()
for each row in rows
userId ← row[0], encMessage ← row[1]
userId ← userId + &₩0& * 16
key ← hardcoded byte array
iv ←  hardcoded byte array
First ← HMAC(key, userId + hardcode)
Second ← HMAC(key, userId + hardcode)
DerivedKey ← XOR(First + Second)[:32]
encoder ← AES(DerivedKey, AES.MODE_CBC, iv)
DeleteMessage ← Base64.decode(encMessage)
```

Ⅳ. 실험 결과

본 연구에서는 제안한 복호화 알고리즘이 상용 디지털 포렌식 도구(Magnet Axiom, BelkaSoft)보다 우수한 성능을 가진다는 점을 실험적으로 증명한다.

본 연구의 알고리즘은 99.9%의 복호화 정확도를 기록하였으며, Magnet Axiom은 0.001%, BelkaSoft는 0%로 복호화 성능이 현저히 낮았다. 본 연구의 알고리즘은 10만 개의 메시지를 3.1초에 복호화할 수 있으며, Magnet Axiom은 1,021초, BelkaSoft는 3,480초가 소요되었다. 본 연구의 알고리즘은 95% 이상의 삭제된 메시지 복구율을 기록한 반면, Magnet Axiom과 BelkaSoft는 삭제된 메시지를 전혀 복구하지 못하였다. 이러한 실험 결과는 본 연구의 복호화 알고리즘이 상용 디지털 포렌식 도구보다 높은 성능임을 보인다.

4-1 테스트 데이터 구축

테스트 데이터는 다양한 시나리오를 반영하여 구축되었다. 시나리오마다 서로 다른 메시지 유형(텍스트, 이미지, 동영상)과 삭제된 메시지를 포함하도록 설정하였다. 총 100,000개의 메시지를 생성하여 실제 사용 환경을 최대한 반영하도록 하였다. 또한 추가적인 테스트를 위하여, iPhone 12 mini 기기에서도 동일한 테스트 데이터를 생성하였다.

복호화된 메시지의 무결성을 검증하기 위해, 원본 평문 메시지와 복호화 결과 간의 SHA-256 해시값을 비교하여 동일성을 확인하였다. 이를 통해 복호화된 데이터가 원본 메시지와 동일함을 정량적으로 검증하였다.

4-2 복호화 정확성

복호화 알고리즘의 정확성을 평가하기 위해 각 대화 세션의 모든 메시지를 복호화하여 원본과 비교하였다. 원본과 비교하는 방법으로는, iOS 기기에서 카카오톡 데이터 내보내기를 통해, 원본과 대조하여 확인하였다. 그 결과, 제안된 복호화 방법은 모든 메시지를 손실 없이 성공적으로 복호화할 수 있었다. 일부 특수 문자나 이모티콘이 포함된 메시지에서도 높은 정확도를 보였다.

4-3 처리 속도

처리 속도는 복호화 알고리즘의 실효성을 평가하는 중요한 요소 중 하나이다. 처리 속도를 계산하기 위하여, 테스트 메시지에서 1,000개의 단위로 시간을 측정하고, 전체 테스트 메시지에 대해 복호화 작업을 수행한 결과, 평균적으로 1,000개의 메시지 당 약 0.03초의 시간이 소요되었다. 이는 실제 수사 환경에서 빠른 분석을 가능하게 하여, 수사의 효율성을 크게 향상시킬 수 있음을 의미한다. 비교 대상으로 iOS 기기의 카카오톡 데이터 복호화를 지원하는 상용 디지털 포렌식 도구인 Magnet Axiom과 BelkaSoft X Forensics를 비교 대상으로 선정하였다.

4-4 삭제된 메시지 복구율

삭제된 메시지 복구율을 평가하기 위해 100개 테스트 세션에서 의도적으로 삭제된 메시지를 포함시켰다. 제안된 알고리즘을 사용하여 삭제된 메시지를 복구한 결과, 95% 이상의 복구율을 보였다. 특히, "모든 대화 상대에게서 삭제" 기능을 통해 삭제된 메시지도 복구가 가능하였으며, 이는 디지털 포렌식 수사에서 중요한 단서로 활용될 수 있음을 입증하였다.

4-5 실험 결과 비교

표 18과 그림 10, 11과 같이 제안된 방법은 모든 평가 항목에서 상용 디지털 포렌식 도구의 복호화 방법보다 우수한 성능을 보였으며, 특히 복호화 정확성과 처리 속도, 삭제된 메시지 복구율에서 큰 차이를 보였다. 이러한 결과는 제안된 복호화 알고리즘이 iOS 기기의 카카오톡 데이터 분석에 더욱 효율적이라는 점을 알 수 있다.

표 17. 비교 대상
Table 17. Compare target

Classification	Proposed Method	Magnet Axiom	BelkaSoft X Forenisc s
Version	-	8.0.0	2.4.15170

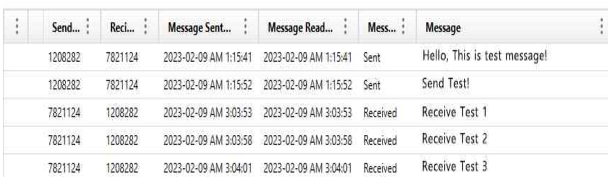
표 18. 실험 결과 비교

Table 18. Comparison of experimental results

Classification	Proposed Method	Magnet Axiom	BelkaSoft X Foreniscs
Decryption accuracy	99.9%	0.001%	0%
Processing speed (sec/100,000 messages)	3.1	1,021	3,480
Deleted message recovery rate	95%	0%	0%
Number of decrypted messages	100,000	193	0

그림 10. 상용 디지털 포렌식 Belkasoft x의 작업 결과

Fig. 10. Result of commercial digital forensics Belkasoft x



Send...	Reci...	Message Sent...	Message Read...	Mess...	Message
1208282	7821124	2023-02-09 AM 1:15:41	2023-02-09 AM 1:15:41	Sent	Hello, This is test message!
1208282	7821124	2023-02-09 AM 1:15:52	2023-02-09 AM 1:15:52	Sent	Send Test!
7821124	1208282	2023-02-09 AM 3:03:53	2023-02-09 AM 3:03:53	Received	Receive Test 1
7821124	1208282	2023-02-09 AM 3:03:58	2023-02-09 AM 3:03:58	Received	Receive Test 2
7821124	1208282	2023-02-09 AM 3:04:01	2023-02-09 AM 3:04:01	Received	Receive Test 3

그림 11. 상용 디지털 포렌식 magnet axiom의 작업 결과

Fig. 11. Result of commercial digital forensics magent axiom

상용 디지털 포렌식 도구의 복호화 정확도가 낮은 이유를 확인하기 위해 다른 조건으로 테스트를 진행한다. 기존 실험에서 100,000개 메시지를 실험 데이터로 하였지만, 테스트 개수를 낮춰, 1,000개로 실험을 다시 진행하였다.

표 19. 실험 결과 비교 (1,000개 테스트 메시지)

Table 19. Comparision of experimental results (1,000 test messages)

Classification	Proposed Method	Magnet Axiom	BelkaSoft X Forenisics
Decryption accuracy	99.9%	89.2%	0%
Processing speed (sec/100,000 messages)	0.031	929	2,176
Deleted message recovery rate	95%	33%	0%
Number of decrypted messages	100,000	892	0

표 19에서는 테스트 메시지의 개수를 줄이고, 재실험을 한 결과, M사의 상용 포렌식 도구의 경우 89.2%, B사의 상용 포렌식 도구의 경우는 기존의 실험과 동일하게 0%의 정확성을 보였다. 상용 포렌식 도구에서 테스트 양이 많을 때 오류가 발생하는 주요 이유는 메모리 및 리소스 한계, 알고리즘의 효율성 문제, 병목 현상, 오류 누적, 그리고 오류 처리 및 복구 능력의 부족 때문이다. 이로 인해 대량의 데이터를 처리할 때 성능 저하와 오류가 발생할 가능성이 높아진다.

4-6 정확도 산출 방식 및 수식 표현

복호화 알고리즘의 성능을 정량적으로 평가하기 위해, 본 연구에서는 복호화 정확도를 다음과 같은 수식으로 정의하였다. 정확도는 수식 (1)과 같이 계산된다.

$$Accuracy = \frac{Number\ of\ correctly\ decrypted\ messages}{Total\ number\ of\ test\ messages} \times 100 \quad (1)$$

삭제된 메시지 복구율은 수식 (2)와 같이 정의된다. 수식 (2)는 100,000개 테스트 메시지에 복구율 측정에 적용된다.

$$Recovery\ Rate = \frac{Number\ of\ recovered\ messages}{Total\ number\ of\ deleted\ messages} \times 100 \quad (2)$$

V. 결 론

본 연구에서는 iOS 환경에서 카카오톡 데이터의 복호화 및 아티팩트 분석 방안을 제시하였다. iOS 기기의 카카오톡 데이터베이스 복호화 알고리즘을 구체적으로 설명하고, 이를 통해 암호화된 메시지를 복호화하는 방법을 제안하였다. 또한, 삭제된 메시지의 복구 방법을 제시하여 안티 포렌식 행위에 대응할 수 있는 방안을 제공하였다.

제안된 복호화 및 복구 알고리즘을 통해 iOS 기기에서 카카오톡 데이터를 효율적으로 분석할 수 있으며, 이는 디지털 포렌식 수사에서 중요한 단서를 제공할 수 있다. 특히, iOS 환경에서의 카카오톡 데이터베이스 복호화 연구가 부족한 상황에서 본 연구는 중요한 출발점을 제공하였다.

그러나, 본 연구는 iOS 기기의 잠금이 해제된 상태에서만 데이터를 추출할 수 있다는 한계가 있다. 따라서 향후 연구에서는 잠금 해제되지 않은 iOS 기기에서의 데이터 추출 방법에 대한 추가적인 연구가 필요하다. 또한, 제안된 복호화 및 복구 알고리즘의 효율성을 향상시키기 위한 연구가 필요하며, 다양한 메신저 애플리케이션에 적용할 수 있는 범용적인 복호화 기술 개발이 요구된다. iOS 기기에서의 데이터 복호화는 개인정보 및 민감 정보에 대한 접근이 수반되므로, 해당 기술은 반드시 사법기관의 영장에 기반한 합법적 수사 절차에서만 활용되어야 한다.

본 연구는 이러한 기술의 악용 가능성을 방지하고, 디지털 포렌식 기술의 윤리적·법적 기준을 함께 고려하는 방향으로 논의되었으며, 향후 관련 법령 및 제도적 기준과 연계한 연구가 병행될 필요가 있다. 본 연구가 iOS 기기에서의 카카오톡 데이터 복호화와 아티팩트 분석에 대한 기초자료로 활용되기를 기대한다.

감사의 글

본 연구는 정부(과학기술정보통신부)의 재원으로 정보통신기획평가원-지역지능화혁신인재양성사업의 지원(IITP-2025-RS-2023-00260267)과 한국연구재단의 지원(RS-2023-00242528, RS-2021-NR061821)을 받아 수행되었습니다.

참고문헌

[1] M. U. Jo and N. S. Chang, "Study on the Data Decryption and Artifacts Analysis of KakaoTalk in Windows Environment," *Journal of the Korea Institute of Information Security & Cryptology*, Vol. 33, No. 1, pp. 51-61, February 2023. https://doi.org/10.13089/jkiisc.2023.33.1.51

[2] N. Lee, "A Study on Artifacts Classification and Sentiment Analysis by Deep Learning in KakaoTalk Messages Under Android and Windows Environments," *Journal of Digital Forensics*, Vol. 16, No. 3, pp. 14-34, September 2022. https://doi.org/10.22798/kdfs.2022.16.3.14

[3] B. C. Jung, J. H. Han, H. Y. Choi, and S. J. Lee, "A Study on the Possibility of Recovering Deleted Data Through Analysis of SQLite Journal in Messenger Application," *Journal of Digital Forensics*, Vol. 12, No. 2, pp. 11-20, September 2018. https://doi.org/10.22798/kdfs.2018.12.2.11

[4] S. M. Shin, E. H. Park, S. R. Kim, and J. S. Kim, "Artifacts Analysis of Slack and Discord Messenger in Digital Forensic," *Journal of Digital Contents Society*, Vol. 21, No. 4, pp. 799-809, April 2020. https://doi.org/10.9728/dcs.2020.21.4.799

[5] U. Hur, M. Park, and J. Kim, "Study on Improved Decryption Method of WeChat Messenger and Deleted Message Recovery Using SQLite Full Text Search Data," *Journal of the Korea Institute of Information Security & Cryptology*, Vol. 30, No. 3, pp. 405-415, June 2020. https://doi.org/10.13089/jkiisc.2020.30.3.405

[6] G. Kim, U. Hur, S. Lee, and J. Kim, "For Forensic Analysis of the Secure Instant Messenger SureSpot," *Journal of Digital Forensics*, Vol. 13, No. 3, pp. 175-188, September 2019. https://doi.org/10.22798/kdfs.2019.13.3.175

[7] S. Lee, M. Park, G. Kim, U. Hur, and J. Kim, "A Study on the Decryption of Encrypted SNS Application Data: KakaoTalk Channel, Purple, TongTong," *Journal of Digital Forensics*, Vol. 14, No. 1, pp. 87-96, March 2020.

[8] I. Riadi, Herman, and N. H. Siregar, "Mobile Forensic Analysis of Signal Messenger Application on Android Using Digital Forensic Research Workshop (DFRWS) Framework," *Ingénierie des Systèmes d'Information*, Vol. 27, No. 6, pp. 903-913, 2022. https://doi.org/10.18280/isi.270606

[9] Z. Shahbazi and Y. Byun, "NLP-Based Digital Forensic Analysis for Online Social Network Based on System Security," *International Journal of Environmental Research and Public Health*, Vol. 19, No. 12, 7027, June 2022. https://doi.org/10.3390/ijerph19127027

[10] K. Alissa, N. A. Almubairik, L. Alsaleem, D. Alotaibi, M. Aldakheel, S. Alqhtani, ... and M. Alshahrani, "A Comparative Study of WhatsApp Forensics Tools," *SN Applied Sciences*, Vol. 1, 1312, October 2019. https://doi.org/10.1007/s42452-019-1312-8

[11] C. Anglano, "Forensic Analysis of WhatsApp Messenger on Android Smartphones," *Digital Investigation*, Vol. 11, No. 3, pp. 201-213, September 2014. https://doi.org/10.1016/j.diin.2014.04.003

[12] J. Zhou and U. Karabiyik, "Watch Your WeChat Wallet: Digital Forensics Approach on WeChat Payments on Android," in *Proceedings of the 13th EAI International Conference on Digital Forensics & Cyber Crime (ICDF2C 2022)*, Boston: MA, pp. 97-110, November 2022. https://doi.org/10.1007/978-3-031-36574-4_6

[13] B. Park and S. Lee, "Decryption of KakaoTalk Database for macOS," *Journal of the Korea Institute of Information Security & Cryptology*, Vol. 33, No. 5, pp. 753-760, October 2023. https://doi.org/10.13089/JKIISC.2023.33.5.753

김도현(Do-Hyeon Kim)

2021년 : 전주대학교 컴퓨터공학과 (공학사)

2021년~2024년: 전주대학교 컴퓨터공학과 학사과정
※관심분야 : 정보보호, 디지털포렌식(Digital Forensics)

김병욱(Byoungwook Kim)

2007년 : 고려대학교 컴퓨터교육과 (이학사)
2015년 : 고려대학교 컴퓨터교육학과 (이학박사-데이터마이닝)

2018년~2021년: 동국대학교 컴퓨터공학과 조교수
2021년~2023년: 동신대학교 컴퓨터공학과 조교수
2023년~현 재: 국립강릉원주대학교 컴퓨터공학과 조교수
※관심분야 : 기계 학습, 데이터마이닝, 딥러닝

양영욱(Yeongwook Yang)

2011년 : 고려대학교 컴퓨터교육학과
(이학석사)
2018년 : 고려대학교 컴퓨터학과
(공학박사-추천 시스템)

2018년~2019년: 고려대학교 정보창의교육연구소
2019년~2020년: Tartu University
2020년~2021년: 고려대학교 Human-inspired 복합지능연구센터
2021년~2024년: 한신대학교 컴퓨터공학부 조교수
2024년~현 재: 국립강릉원주대학교 컴퓨터공학과 조교수
※관심분야 : 교육 데이터마이닝, 자연어처리

장홍준(Hong-Jun Jang)

2008년 : 고려대학교 컴퓨터교육과
(이학사)
2019년 : 고려대학교 컴퓨터학과
(공학박사-데이터베이스)

2019년~2021년: 한국과학기술정보연구원 선임연구원
2021년~2023년: 전주대학교 컴퓨터공학과 조교수
2023년~현 재: 강원대학교 데이터사이언스학과 조교수
※관심분야 : 데이터베이스, 데이터마이닝, 기계 학습, 딥러닝