Journal of The Korea Institute of Information Security & Cryptology ISSN 1598-3986(Print)
VOL.35, NO.6, Dec. 2025 ISSN 2288-2715(Online)
https://doi.org/10.13089/JKIISC.2025.35.6.1349

플랫폼 환경에 따른 온라인 메신저 포렌식의 체계적 비교 분석 연구*

이 수 연,[1†] 샤흐저드,[1] 구 형 준[2‡]
[1,2]성균관대학교 (대학원생, 교수)

Online Messenger Forensics Across Platforms: A Systematic Comparative Study*

Suyeon Lee,[1†] Shakhzod Yuldoshkhujaev,[1] Hyungjoon Koo[2‡]
[1,2]Sungkyunkwan University (Graduate Student, Professor)

요 약

오늘날 온라인 메신저는 다양한 플랫폼에서 광범위하게 사용하는 주요 의사소통 도구다. 사용자 행위에 따라 생성되는 디지털 흔적은 디지털 포렌식 분석에서 핵심적인 단서를 제공한다. 그러나 운영 체제, 저장 구조, 암호화 기법의 차이로 인해 메신저별 데이터 보존 방식과 수집 가능 정보가 상이하며, 이에 따라 분석 전략 또한 달라진다. 본 연구는 주요 메신저 15종을 대상으로 저장 위치, 보안 구조, 아티팩트 유형, 분석 도구, 수집 절차를 체계적으로 정리하고 비교한다. 플랫폼 환경에 따른 아티팩트의 위치와 특성을 분석하고, 도구와 수집 방식의 조합이 데이터 복구 범위에 미치는 영향을 평가한다. 또한 서비스별 분석 접근 방식을 분류하고 기술적 제약 요인을 식별함으로써 환경에 따른 분석 절차을 구조화한다. 이를 통해 메신저 포렌식 분야에서 일관된 분석 절차 수립과 자동화 도구 설계의 필요성을 도출하고, 다양한 플랫폼과 조건에 적용 가능한 전략을 수립할 수 있는 토대를 제공한다.

ABSTRACT

Instant messengers have become essential communication tools, widely used across varying platforms. The digital traces from user activities provide critical evidence for forensic investigations. However, differences in operating systems, storage architectures, and encryption techniques lead to variations in how data is preserved and what information can be collected from each messenger, which requires forensic investigation strategies tailored to each service. This study systematically compares 15 major messengers by examining storage locations, security mechanisms, artifact types, forensic tools, and data collection procedures. It analyzes the location and characteristics of artifacts based on platform environments, evaluating how combinations of tools and collection methods affect the scope of data recovery. By categorizing service-specific approaches and identifying technical constraints, we establish a forensic workflow adaptable to different platform environments. These findings necessitate standardized forensic procedures and the development of automated tools. Ultimately, we aim to design versatile analysis strategies applicable across a wide range of platforms and operational conditions.

Keywords: Online Messenger Services, Messenger Forensics, Artifact Analysis, Digital Evidence

Received(08. 07. 2025), Modified(11. 03. 2025), Accepted(11. 21. 2025)

* 본 논문은 2025년도 한국정보보호학회 하계학술대회에 발표한 우수논문을 개선 및 확장한 것임

* 본 논문은 2025년도 정부(과학기술정보통신부) 정보통신기획평가원 No.RS-2024-00398745, 디지털 환경에서의 증거인멸행위 증명 및 대응기술 개발) 지원으로 수행한 연구임.

† 주저자, susan2ee@g.skku.edu
‡ 교신저자, kevin.koo@g.skku.edu(Corresponding author)

I. 서 론

메신저 서비스는 개인 간 소통 수단을 넘어 기업 내부 커뮤니케이션 및 사이버 범죄의 주요 플랫폼으로 자리매김함에 따라, 이들 서비스가 생성하는 디지털 흔적은 법 집행 기관과 보안 전문가에게 필수적인 증거 자료로 활용되고 있다[1]. KakaoTalk[9], Telegram[15] 등 주요 메신저는 메시지 전송 기록, 첨부 파일, 로그인 이력 등을 로컬 장치나 클라우드에 저장하며, 이 과정에서 남은 아티팩트(artifact)는 범죄 수사 및 법정 증거 확보에 결정적 역할을 수행한다. 그러나 각 플랫폼이 제공하는 저장 구조, 암호화 정책, 백업 방식의 차이로 인해 포렌식 분석 전략은 상이하며, 동일한 복구 방법을 적용하더라도 복구 가능한 데이터 범위와 정확도에서 큰 편차가 나타난다.

일례로, Telegram[15]은 도난 데이터 거래를 비롯한 다양한 불법 활동의 핵심 플랫폼으로 악용되고 있으며, 최근 연구에서는 메신저 채널의 운영 방식과 거래 절차를 범죄 시나리오 관점에서 분석하고 이에 기반한 예방 전략을 제시하였다[2]. NDTA 2024는 카르텔이 WhatsApp[18], Signal[14] 등 암호화 메신저를 통해 광고, 결제, 고객 응대를 주기적으로 온라인화했으며, 실제 수사에서도 그 활용이 광범위함을 보여준다. 이에 따라 범죄 사건의 원인 규명을 위해서는 메신저 포렌식을 통해 디바이스에서 추출한 대화, 미디어, 연락처, 그리고 위치 데이터 등이 핵심 증거수단으로 활용할 수 있다[3]. 이처럼 메신저 포렌식은 단순 증거 회수 단계를 넘어, 플랫폼별 기능과 정책에 따른 디지털 아티팩트를 고려해 수사할 필요가 있다.

기존 연구들은 주로 개별 메신저에 대한 아티팩트 복구 기법을 사례별로 보고했으나, 플랫폼 지원 범위, 보안 기능, 저장 위치에 따른 통합적 비교 분석은 부족한 실정이다. 본 논문은 지난 10년간 발표된 포렌식 연구를 대상으로 15종의 주요 메신저에 대해 실험 환경, 저장 구조, 주요 복구 대상 아티팩트, 사용된 분석 도구 현황을 체계적으로 정리 비교함으로써, 플랫폼별 복구 가능성의 차이와 기술적 한계를 종합적으로 분석한다. 이를 통해 연구자와 수사관이 각 메신저 특성에 최적화된 포렌식 전략을 수립할 수 있는 토대를 제공하고자 한다.

II. 온라인 메신저 포렌식

2.1 배경 개념 및 주요 용어

온라인 메신저 포렌식(messenger forensics)은 메신저 서비스 이용 중 생성되는 다양한 디지털 흔적을 수집하고 분석하여, 법적 증거로 활용 가능한 형태로 가공하는 과정을 의미한다. 최근 메신저는 단순한 소통 수단을 넘어 기업 내부의 협업 도구, 공공기관의 행정 수단, 그리고 사이버 범죄의 통신 채널로까지 활용되는, 그 사회적 활용도가 빠르게 확장되고 있다. 이러한 흐름에 따라, 메신저 포렌식은 디지털 포렌식 분야 내 독립된 분석 영역으로 주목받고 있으며, 메신저가 남기는 디지털 흔적은 수사기관과 보안 전문가에게 중요한 단서로 기능하고 있다[4].

메신저 포렌식에서 가장 핵심인 아티팩트는 메신저 사용 과정에서 생성된 데이터 조각으로, 포렌식 도구로 확인 가능하다. 예컨대, 로컬 DB(SQLite, JSON 등), SharedPreferences, 로그 파일, 캐시 디렉토리, 네트워크 트래픽, 메모리 내 임시 객체 등이 포함된다. 이런 아티팩트는 적절한 분석과 해석 과정을 거쳐 맥락이 부여되었을 때 비로소 증거(evi-dence)로서의 의미를 지닌다. 예를 들어, 메신저 데이터베이스에서 추출한 타임스탬프는 그 자체로는 단순한 숫자에 불과하지만, 시스템 시간대 정보 및 사용자의 위치 기록과 결합되면 특정 시점의 행위를 입증하는 증거로 기능할 수 있다. 또한 삭제된 메시지 레코드도 WAL(Write-Ahead Logging) 파일이나 저널 파일과 교차 분석함으로써 실제 대화 내용과 삭제 시점을 재구성할 수 있다. 또한, 아티팩트는 저장 환경에 따라 휘발성(volatile) 정보와 비휘발성(non-volatile) 정보로 구분된다. 휘발성 정보는 디바이스의 전원이 꺼지면 소멸하는 메모리 기반의 데이터로, 예를 들어 앱 실행 중 생성된 세션 토큰이나 인증 정보, 실시간 채팅 로그 등이 이에 해당한다. 반면 비휘발성 정보는 디스크에 영구 저장되는 자료로, 로컬 DB, 첨부 파일, 삭제되지 않은 캐시 파일, 시스템 로그 등이 이에 포함된다. 포렌식 분석에서는 이러한 정보의 특성을 파악하고, 보존 상태에 따라 복구 전략을 설정해야 한다.

2.2 일반적인 메신저 포렌식 절차와 분석 대상

메신저 포렌식은 디지털 포렌식의 표준 절차인 수집

(collection), 보존(preservation), 분석(analysis), 보고(reporting) 4단계 프레임워크를 따른다[5]. 다만 각 단계에서는 메신저 서비스의 구조적 특성(암호화 방식, 저장 구조, 플랫폼별 접근 제약 등)을 고려한 세부 전략이 요구된다. 첫째, 수집 단계에서는 분석 대상 장치나 서버로부터 메신저 관련 데이터를 확보한다. 확보 대상에는 채팅 기록뿐 아니라 첨부 파일, 세션 정보, 인증 토큰, 로그인 이력, 미디어 캐시, 네트워크 트래픽 로그 등이 포함된다. 메신저 포렌식에서는 특히 모바일 환경의 운영체제(Android/iOS) 및 루팅 여부에 따라 접근 가능한 파일 시스템 영역과 아티팩트 범위가 제한되므로, 물리적 덤프(physical dump)와 논리적 추출(logical extraction) 방식의 선택이 분석 결과에 직접적인 영향을 미친다. 둘째, 보존 단계에서는 확보된 데이터의 무결성을 유지하기 위한 조치가 수행된다. 대표적인 방법으로는 해시값(SHA-256 등)의 생성, 원본 및 복사본의 분리 보관, 메타데이터 캡처 등이 있으며, 이는 이후 법정 제출 과정에서 증거물의 신뢰성을 입증하기 위한 기반이 된다. 셋째, 분석 단계에서는 수집한 데이터를 기반으로 사용자 행위를 추론하고, 필요한 경우 삭제되거나 변형된 데이터를 복구한다. 메신저의 경우 SQLite 데이터베이스의 WAL 파일, JSON 기반 설정 파일, 그리고 암호화된 로컬 스토리지 등 서비스마다 상이한 저장 포맷을 다뤄야 하므로, 메타데이터 분석, 로그 상관 분석, 데이터 카빙(data carving) 등의 기법이 플랫폼 특성에 맞게 적용된다. 마지막으로, 보고 단계에서는 분석 결과를 법적으로 유효한 형식으로 정리한다. 사용된 도구, 복구 결과, 판단의 근거 등을 상세히 기록하고, 동시에 증거 보관의 연속성(chain of custody)을 명확히 남김으로써 증거의 신뢰성과 법적 효력을 뒷받침한다.

이와 같은 절차는 모든 메신저 포렌식 작업의 공통된 틀을 구성하며, 분석 대상이 되는 플랫폼의 구조적 특성과 보안 정책, 데이터 저장 방식에 따라 적용 방법과 도구는 달라질 수 있다.

III. 분석 프레임워크

메신저 포렌식은 단순한 데이터 복구 작업을 넘어, 다양한 플랫폼 환경과 보안 정책에 따라 달라지는 아티팩트 특성을 구조적으로 이해하고 분석하는 과정을 포함한다. 본 절에서는 기존의 대표적인 포렌식 연구들을 검토하여, 메신저 분석에서 일반적으로 사용되는 절차를 세 단계로 정리하였다. Fig. 1.은 다수의 선행 연구에서 제시된 실험 설계와 분석 절차 전반을 정리한 것으로, 사용자 행위 시나리오 정의(scenario definition), 디지털 증거 수집(data acquisition), 디지털 증거 분석(artifact analysis)의 3단계로 구성된다. 각 단계의 세부 내용은 이후 3.2~3.4절에서 상세히 다룬다.

3.1 메신저 서비스 분류 기준

메신저 포렌식 분석의 체계화를 위해, 플랫폼, 저장 구조, 보안 정책의 세 가지 분류 기준을 제시한다. 첫째, 플랫폼 측면에서는 모바일 환경(Android/iOS), Windows 및 macOS 기반의 데스크톱 클라이언트, 그리고 웹 브라우저 기반의 웹 클라이언트로 구분된다. 각 플랫폼은 파일 접근 권한, 메모리 관리 방식, 아티팩트 생성 및 저장 위치에 영향을 미친다. 둘째, 저장 구조는 애플리케이션에서 생성되는 SQLite 데이터베이스나 SharedPreferences, pli-st 형식의 설정 파일, 파일 시스템 내 로그 및 캐시

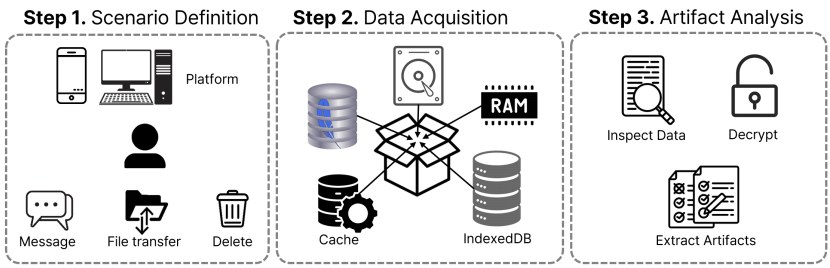

Fig. 1. Framework for the digital forensics of online messengers. Three-step process: scenario definition, data acquisition from various sources, and artifact analysis for evidence extraction.

디렉토리, 그리고 브라우저의 LocalStorage, Inde-xedDB 등으로 나뉜다. 이러한 저장 구조는 아티팩트의 식별 가능성과 분석 난이도를 결정짓는 요소로 작용한다. 셋째, 보안 정책은 종단간 암호화(End-to-End Encryption, E2EE) 적용 여부, 임시 저장 메시징 기능(ephemeral messaging), 클라우드 백업 방식 등을 포함한다. 이는 동일한 행위가 이루어지더라도 복구 가능한 데이터의 범위와 형식에 중대한 영향을 미치며, 메신저별 분석 전략을 설계하는 데 핵심적인 기준이 된다. 이러한 분류 기준은 이후 사용자 행위 시나리오 정의, 증거 수집 및 분석 단계에서 각 메신저의 특성을 구조화하고 비교 분석하는 데에 활용된다.

3.2 사용자 행위 시나리오 정의

메신저 포렌식 실험 환경에서는 Windows, Android, iOS, 웹(Web) 등 주요 플랫폼에 메신저 애플리케이션을 설치한 후, 실제 사용자가 수행할 수 있는 모든 동작을 모사하는 사용자 행위 시나리오를 설계한다. 본 연구의 시나리오 설계는 기존 메신저 포렌식 연구[52,62,63]에서 공통적으로 사용된 기본 행위 패턴(계정 생성, 메시지 송수신, 삭제 및 복구)을 기반으로 하되, 실제 범죄 수사 및 포렌식 실무에서 빈번히 요구되는 복구 시나리오를 추가로 반영하였다. 이 시나리오는 계정 생성 및 삭제, 텍스트 및 파일 송수신, 채팅방(또는 채널) 생성 및 탈퇴, 메시지나 파일 삭제 및 복구 시도 등을 포함하며, 임시 저장 메시지 기능, 애플리케이션 재설치, 로그아웃 후 재로그인과 같이 복구 난이도가 높은 예외적 조건도 함께 반영한다. 각 시나리오는 플랫폼별 저장 구조와 아티팩트 특성 차이를 비교, 분석하기 위한 기준이 되며, 범죄 수사 상황을 가정한 메시지 은닉, 계정 우회 접근과 같은 의심 행위까지 포함하도록 설계한다(Table 1.).

3.3 디지털 증거 수집

정의된 시나리오를 바탕으로 실제 디바이스나 에뮬레이터 상에서 사용자 행위를 수행한 후, 다양한 저장소로부터 증거를 획득한다. 애플리케이션의 로컬 저장소에서는 SQLite 데이터베이스 파일, 캐시 디렉토리 내 미디어, 로그 파일, 파일 시스템에 기록된 설정, 로그 데이터를 수집하며, 웹 클라이언트의 경우 IndexedDB나 LocalStorage 등을 추출한다. 또한, 시스템 메모리(RAM) 덤프를 통해 휘발성 데이터를 확보하고, 필요시 네트워크 트래픽 패킷을 캡처하여 클라우드 동기화 흔적을 분석하기도 한다. 이때 USB 디버깅(ADB), 포렌식 이미징(FTK Imager), 논리 추출 도구 등 각 저장소 특성에 적합한 방법을 선택하여 물리적 이미징, 메모리 덤프, 논리 추출 등으로 증거를 확보하며, 해시 기반 무결성 검증 절차를 거쳐 원본성 보장을 원칙으로 한다.

Table 1. Structured categorization of online messenger forensic artifacts by user activity type in the existing forensic literature

	Activity Type	Description
Account	Creation	Artifacts from creating an account
	Deletion	Remnant data after account removal
	Authentication	Session tokens or login/logout records
Message	Text transmission	Metadata of text messages that are sent or received
	Media transfer	Storage and logs of file attachments(images, videos, etc.)
	Group participation	Artifacts from joining or leaving group chats or channels
	Deletion	Traces after message or file removal(local/cloud)
	Ephemeral messaging	Temporary data from disappearing message features
Access & Bypass	Device modification	Changes due to rooting or jailbreaking
	Lock bypass	Logs from screen unlocking bypass attempts
Anti-Forensic	Reinstallation	Attempting to delete residual data with app reinstallation
	Cache removal	Attempting to erase artifacts by clearing logs or caches

3.4 디지털 증거 분석

수집된 원시 데이터는 채팅 기록, 파일 전송 내역, 계정 정보, 세션 토큰, 삭제된 메시지 등 다양한 아티팩트를 포함한다. 분석 단계에서는 SQL 쿼리 및 스크립트를 이용해 데이터베이스를 직접 파싱하고, 메모리 검색 기법을 활용해 휘발성 키와 토큰을 추출한다. 암호화가 적용된 경우에는 SQLCipher 복호화나 E2EE 해제 절차를 수행하며, 삭제된 파일, 레코드는 바이트 시그니처 기반 데이터 카빙 기법으로 복원한다. 이후 복원된 아티팩트를 유형별, 플랫폼별로 구조화하여, 저장 위치, 복구 가능성, 암호화 적용 여부 등을 종합적으로 평가함으로써 메신저별 포렌식 특성을 체계적으로 파악한다.

IV. 분석 환경 및 활용 도구 비교

본 연구는 지난 10년간 발표된 메신저 포렌식 관련 논문을 조사하여, Telegram[15], Signal[14], KakaoTalk[9] 등 포렌식 분석 빈도가 높고, 사용자 수가 많은 메신저 애플리케이션을 중심으로 총 15종의 주요 메신저를 비교 대상으로 선정했다. 선정 기준은 국내외 사용자 기반, 포렌식 분석 보고 빈도, 다양한 플랫폼에서의 실행 가능성을 반영했다. 각 메신저가 실행되는 플랫폼별 분석 환경과, 연구에서 활용된 대표적인 포렌식 도구를 살펴본다.

4.1 분석 환경

메신저 포렌식 연구에서 사용된 분석 환경은 각 메신저의 플랫폼 지원 여부에 따라 구성된다. 일반적으로 Android 및 Windows 환경이 가장 널리 활용되며, 모바일 분석에는 실제 기기와 에뮬레이터(iOS 시뮬레이터 포함)가 병행된다. Android 및 iOS 환경에서는 애플리케이션 데이터베이스, 캐시, 설정 파일이 저장되는 경로와 접근 권한 구조를, 데스크톱 환경(Windows, macOS, Linux)에서는 운영체제별 표준 데이터 디렉토리 및 사용자 설정 저장소를 주요 대상으로 본다. 웹 클라이언트를 분석할 때는 Chrome, Firefox, Edge와 같은 여러 브라우저 환경을 기반으로 조사하며, IndexedDB, LocalStorage 등 브라우저 저장소에 남는 대화 및 미디어 흔적이 포함한다. 환경별 저장 구조 차이가 디지털 아티팩트의 생성, 보존, 복구 과정에 영향을 미치기 때문에 Table 2.에서 조사 대상 15종 메신저의 플랫폼별 지원 여부를 정리한다.

Table 2. Platform availability of 15 messenger applications of our choice. Mob, Win, Mac, Lin, and Web denote mobile(Android & iOS), Windows, macOS, Linux, and Web browser, resepectively.

Messengers	Platform				
	Mob	Win	Mac	Lin	Web
Discord [6]	✓	✓	✓	✓	✓
Element [7]	✓	✓	✓	✓	✓
Facebook [8]	✓	✓	✓	×	✓
Kakaotalk [9]	✓	✓	✓	×	×
Line [10]	✓	✓	✓	×	✓
Microsoft Teams [11]	✓	✓	✓	✓	✓
Naver Band [12]	✓	✓	✓	×	×
Slack [13]	✓	✓	✓	✓	✓
Signal [14]	✓	✓	✓	✓	×
Telegram [15]	✓	✓	✓	✓	✓
Threema [16]	✓	✓	✓	×	×
WeChat [17]	✓	✓	✓	×	×
WhatsApp [18]	✓	✓	✓	×	✓
Wickr [19]	✓	✓	✓	✓	×
Wire [20]	✓	✓	✓	✓	×

4.2 포렌식 도구

증거 수집 및 아티팩트 분석 단계에서는 플랫폼 및 저장소 유형별로 다양한 도구가 활용된다. 먼저, 메모리 이미징을 위해 Windows 환경에서 WinPmem, Android 환경에서 LiME(Linux Memory Extractor)를 활용하며, 이후에는 Volatility와 같은 분석 프레임워크를 활용해 메모리 내용을 정밀하게 분석한다. 디스크 이미징을 위해서는 FTK Imager나 리눅스/유닉스 기반의 명령어인 dd 등이 자주 사용된다. 이들은 디스크 전체를 복사하는 데 유용한 도구로 널리 활용된다. 획득한 이미지의 파일 시스템은 Autopsy, X-Ways Forensics 등을 통해 파싱한다. 모바일 애플리케이션의 데이터는 ADB(Android Debug Bridge), iTunes 백업 기능의 논리 추출 기법을 통해 수집한

Table 3. Category of varying digital forensic tools for messengers by target and purpose

Target	Purpose	Tools
Memory	RAM acquisition	Belkasoft Live RAM Capturer [21], FTK Imager [22], WinPmem [23]
	Analysis	RAMAS [5]
Disk	File system parsing	Autopsy [24], Magnet AXIOM [25], Paraben E3 [26]
	Analysis	X-Ways forensics [27]
Mobile	Data acquisition	Oxygen Forensic Detective [28], UFED [29], XRY [30]
	Artifact parsing	aLEAPP [31], Andriller [32], ADB [33]
	Analysis	Physical Analyzer [34], XAMN [35], XEC Director [36], Kiosk [37], Magnet Acquire [38]
SQLite	Data recovery	Bring2lite [39], FQlite [40], SQLECmd [41]
	Analysis	DB Browser for SQLite [42], Forensic Toolkit [43], KS DB Merge Tools [44]
Browser	Data acquisition	Browser History Capturer [45], Browser History Examiner [46]
	Analysis	Browser History Viewer [47]

다. SQLite DB와 설정 파일은 DB Browser for SQLite 또는 Oxygen SQLite Viewer 를 이용해 내부 구조를 분석한다. 웹 저장소를 분석할 때는 Browser History Capturer나 Browser History Viewer 등 특화된 도구가 사용되며, Magnet AXIOM, Paraben E3 같은 통합 포렌식 툴은 분석 자동화 및 보고서 생성에 유용하다. 이러한 도구들은 각 메신저의 저장 방식 및 보안 정책 차이에 유연하게 대응하는데 필요한 기술적 기반을 제공한다. Table 3.은 유형별 대표 포렌식 도구와 활용 목적을 정리한 것이다.

V. 메신저별 포렌식 아티팩트 비교 분석: 플랫폼, 저장 구조, 보안 정책 관점

본 절에서는 플랫폼, 저장 구조, 보안 정책의 세 가지 기준에 따라 각 메신저의 분석 대상 및 복구 가능 아티팩트를 조사하였다. Table 4.는 최근 10개년간 발표된 연구를 종합하여 메신저별 분석 플랫폼과 복구 가능 아티팩트를 체계적으로 분류한다.

5.1 Discord

Discord는 E2EE를 지원하지 않는 것이 공식 정책이며, 모든 메시지는 서버에 평문으로 저장된다[48]. Android 환경에서는 /data/data/com.discord/cache 경로의 JSON 캐시와 SharedPreferences를 통해 메시지 및 사용자 정보를 확인할 수 있으며, 삭제 메시지는 로컬에 저장되지 않는다. Windows에서는 %AppData%\discord\Local Storage\leveldb 경로의 LevelDB와 JSON 구조를 통해 세션 정보와 캐시를 복원할 수 있다. Autopsy, FTK Imager와 같은 포렌식 도구로 해당 경로를 직접 수집하거나, strings 명령어를 통해 평문 토큰과 사용자 ID를 추출할 수 있다. Linux에서는 메모리 덤프를 통해 JSON 객체 형태로 서버 ID, 채널 ID, 메시지 내용을 일부 복원할 수 있다. 2021년 이후 Electron 기반 구조를 유지하고 있어 버전 간 아티팩트 저장 위치가 안정적이나, 2023년 업데이트부터 캐시 정리 정책이 강화되어 휘발성 데이터의 잔존 시간이 단축되었다. 웹 환경에서는 IndexedDB와 LocalStorage에 세션 토큰이 로그아웃 이후에도 잔류하며, 이는 세션 하이재킹의 위험 요소로 작용할 수 있다[49].

5.2 Element

Element는 Matrix 프로토콜 기반의 메신저로, 기본적으로 E2EE를 지원하는 것이 공식 정책이며, Double Ratchet 알고리즘을 구현한 Olm과 대규모

Table 4. Platform-specific forensic investigation for 15 online messenger applications: storage structures, security policies, and research gaps(§ 5.16 provides a comprehensive comparative analysis).

Messenger	Platform	Storage structures	Security policies	Ref.
Discord	Android	JSON cache, SharedPrefs	No E2EE; no recovery; plaintext creds	[48]
	Windows	LevelDB, AppData cache	No E2EE; cache/session data in plaintext	[48]
	Linux	In-memory JSON	No E2EE; residual data persists	[61]
	Web	IndexedDB, LocalStorage, CacheStorage	No E2EE; tokens persist post logout	[49]
Element	Windows	SQLCipher DB, IndexedDB, Local/SessionStorage	E2EE; keys in Credential Manager	[50]
Facebook	Android	SQLite DB, Media cache	No E2EE; deleted messages recoverable	[53,54]
	Web	Cache, IndexedDB	No E2EE; plaintext session	[51,52]
Kakaotalk	Windows	Encrypted .edb DB	No E2EE; deleted messages decryptable	[55,56]
	macOS	SQLCipher DB, Encrypted multimedia	AES-256; partial deleted data recoverable	[55]
Line	Android	Plaintext SQLite, User data	No E2EE; partial recovery varies by tool	[57]
	Web	In-memory prefs, IndexedDB	No E2EE; plaintext creds	[51]
Microsoft Teams	Android	SQLite DB, SharedPrefs	No E2EE; some tokens retrievable	[58,59]
	iOS	SQLite, JSON, Keychain	No E2EE; auth in Keychain	[59]
	Windows	IndexedDB, AppData cache	No E2EE; plaintext logs & chat flags	[58,59]
Naver Band	Android	SQLite DB, XML, Media cache	No E2EE; plaintext tokens; deleted flagged	[60]
Slack	Android	LevelDB(WebView)	No E2EE; minimal deleted data	[48]
	Windows	LevelDB, AppData cache	No E2EE; plaintext artifacts	[48]
	Linux	In-memory JSON	No E2EE; secure memory design	[61]
Signal	Android	Encrypted SQLite DB, Profile images	E2EE; PIN-derived key; deleted may linger	[62]
Telegram	Android	Cache4.db, Userconfig.xml, Media folders	Mixed; plaintext & E2EE(secret chats)	[51,65]
	Windows	Encrypted tdata, in-memory JSON	No local storage; memory only	[64]
	Web	IndexedDB, LocalStorage	No E2EE locally; volatile memory	[51,63]
Threema	Android	Encrypted SQLite DB	E2EE; AES-256; client-side key	[62,66]
WeChat	Android	SQLCipher DB, Media folder	No E2EE; Encrypted DB; secure-delete	[67]
	Windows	SQLCipher DBs, FileStorage	Encrypted DB; journal overwrite	[67]
WhatsApp	Windows	7 SQLite DBs in LocalState, nondb_settings*.dat	SEE encryption; E2EE in transit	[68]
	Web	IndexedDB, LocalStorage	E2EE; AES-128; HKDF; server salt	[68]
Wickr	Android	SQLCipher DB, GCM blobs, Encrypted media, Plaintext logs	E2EE; AES-GCM; auto-login via device ID	[62]
	Windows	SQLCipher DB, .wic key, Encrypted attachments, Plaintext logs	E2EE; AES-CBC attachments	[69]
Wire	Windows	SQLite cookies, LevelDB logs, Plaintext logs	E2EE(default); local session artifacts	[70]

그룹 채팅을 위한 Megolm을 사용한다. Windows 환경에서의 포렌식 분석이 활발히 이뤄졌으며[50], 주요 데이터는 SQLCipher로 암호화된 Events.db에 저장된다. 복호화 키는 Windows Credential Manager에 평문으로 저장되어 있어, cmdkey /list 명령어를 통해 추출이 가능하다. 추출된 키를 이용해 SQLCipher DB를 복호화하면 전체 대화 기록, 사용자 ID, 타임스탬프를 복원할 수 있다. IndexedDB와 LocalStorage에도 일부 채팅 메타데이터가 잔존하며, Cache 데이터에서는 키보드 입력 이벤트나 클릭 로그가 평문으로 식별되기도 한다. Element는 2020년 5월부터 신규 비공개 대화에 대해 E2EE를 기본으로 활성화하는 정책을 적용했으며, 이후 버전에서는 키 저장 및 복구 메커니즘이 강화되었다. 다만 Credential Manager에 키가 평문으로 노출되는 구조적 한계로 인해, 물리적 접근이 가능하면 암호화가 무력화될 수 있다.

5.3 Facebook

Facebook Messenger는 2016년부터 선택적 E-2EE 기능인 "Secret Conversations"를 제공했으나 기본적으로 활성화되지 않았다. 2023년 12월부터 Meta는 모든 개인 메시지와 통화에 대해 E2-EE를 기본으로 적용하는 정책으로 전환을 시작했다고 발표했으나, 이 전환은 점진적으로 진행 중이며 모든 사용자에게 완전히 적용되지 않았다. 다만 본 연구에서 검토한 포렌식 연구[51,52,53,54]는 대부분 이전 버전을 대상으로 수행되어, E2EE가 적용되지 않은 환경에서의 분석 결과를 다룬다. Android 환경에서는 /data/data/com.facebook.orca/ 경로의 SQLite DB와 미디어 캐시를 통해 메시지, 이미지, 타임스탬프를 복원할 수 있으며, 삭제된 메시지도 DB의 언할당 영역이나 WAL 파일에서 복구 가능하다. Autopsy, Belkasoft Evidence Cent-er와 같은 포렌식 도구를 사용하여 SQLite DB를 파싱하거나, sqlite3 명령어로 직접 쿼리할 수 있다. 웹 환경에서는 IndexedDB 및 CacheStorage에 채팅 로그와 세션 정보가 평문으로 저장되며, 브라우저 개발자 도구를 통해 접근 가능하다. 인증 토큰이 LocalStorage에 잔존하여 세션 탈취 위험이 존재한다. 2019년 이후 버전에서는 앱 데이터 백업 제한이 강화되었으나, 루팅된 기기에서는 여전히 전체 데이터베이스에 접근할 수 있다.

5.4 KakaoTalk

KakaoTalk은 2014년 한국 정부의 메시지 압수 논란 이후 선택적 E2EE 기능인 "Secret Chat"을 도입했으나, 기본 채팅 모드는 E2EE가 적용되지 않으며 데스크톱 버전에서는 Secret Chat 기능 자체가 지원되지 않는다. Windows 환경에서는 %AppData%\Kakao\KakaoTalk\users\ 경로에 사용자별 SQLite 및 .edb 파일이 저장되며, 암호화 키가 디렉토리명과 레지스트리 값을 조합하여 생성되는 알고리즘이 알려져 있어 포렌식 도구를 통한 복호화가 가능하다[55,56]. 삭제된 메시지도 DB 파일의 slack space에서 복구할 수 있다. macOS에서는 SQLCipher로 암호화된 DB가 사용되며, 멀티미디어 파일은 AES-256으로 암호화되어 저장된다[55]. 결과적으로 KakaoTalk은 기본 채팅에서 E2EE를 지원하지 않고, 알려진 키 생성 알고리즘으로 인해 포렌식 분석 난이도가 중간 수준에 해당한다. 다만 Secret Chat 사용 시에는 E2EE로 보호되어 분석이 어려워진다.

5.5 Line

Line은 2015년 9월 "Letter Sealing"이라는 선택적 E2EE 기능을 도입했으며, 2016년부터 기본적으로 활성화하는 정책으로 전환했다. 2024년 11월까지 이미지, 비디오, 파일 첨부에도 E2EE 적용 범위가 확대되었다. 그러나 본 연구에서 검토한 연구[51,57]는 대부분 Letter Sealing이 비활성화되었거나 초기 버전을 대상으로 수행되어, E2EE가 적용되지 않은 환경에서의 평문 데이터 분석 결과를 제시한다. Android 환경에서는 /data/data/jp.nave-r.line.android/databases/ 경로의 SQLite D-B에 메시지 및 사용자 정보가 대부분 평문으로 저장된다. DB Browser for SQLite, Autopsy 등의 도구로 naver_line DB를 파싱하거나, ADB를 통해 루팅된 기기에서 직접 추출할 수 있다. 일부 미디어 파일은 /sdcard/Pictures/Line/ 경로에 캐시로 존재하며, 삭제 시에도 잔존하는 경우가 있다. 웹 환경에서는 앱 설치 직후 메모리에 사용자 정보가 일시적으로 존재하며, 비밀번호는 휘발성으로 저장된다[51]. 사용된 포렌식 도구(Belkasoft, Oxygen Forensic, MOBILedit)에 따라 복구 정확도에 편차가 발생하며, 2018년 이후 버전에서는 캐시 정리

정책이 강화되어 삭제 메시지 복구율이 감소했다.

5.6 Microsoft Teams

Microsoft Teams는 기본적으로 TLS와 SRTP를 사용하여 전송 중 데이터를 암호화하지만, E2EE는 Teams Premium 기능으로 2021년 12월부터 1:1 통화에 한해 선택적으로 제공되며 기본적으로 비활성화되어 있다. 따라서 대부분의 일반 사용자 환경에서는 E2EE가 적용되지 않는다. Android, iOS, Windows를 포함한 다양한 운영체제에서 공통적으로 SQLite 기반의 데이터 저장 방식을 사용한다. Android에서는 /data/data/com.microsoft.teams/databases/SkypeTeams.db에 채팅과 설정 정보가 저장되며, 평문으로 접근 가능하다. DB Browser for SQLite, Belkasoft Evidence Center를 통해 DB를 파싱하거나, ADB를 이용한 물리적 덤프로 데이터를 추출할 수 있다. iOS는 Documents 폴더 내 SQLite와 JSON 파일에 데이터가 저장되며, 인증 정보는 Keychain에 암호화되어 저장되는 것으로 추정된다[59]. Windows에서는 %AppData%\Microsoft\Teams\ 경로의 IndexedDB와 Cache에 로그, 채팅 기록, 파일 정보가 존재하며, 삭제 메시지에도 플래그가 남는다[58,59]. 2020년 이후 버전부터는 캐시 구조가 Chromium 기반으로 변경되어 Electron 앱 포렌식 도구를 활용한 분석이 효과적이며, 2022년 업데이트에서는 로그 보존 기간이 단축되었다. E2EE가 적용되지 않아 로컬 아티팩트 복구 가능성이 높다.

5.7 Naver Band

Naver Band는 E2EE를 지원하지 않으며, 기본 채팅 모드에서 평문 또는 전송 계층 암호화만 적용된다. Android 환경에서는 /data/data/com.nhn.android.band/databases/ 경로의 SQLite DB와 XML 파일을 통해 채팅, 밴드 정보, access token 등을 저장한다[60]. DB Browser for SQLite, Autopsy를 사용하여 DB를 파싱하거나, ADB를 통해 루팅된 기기에서 데이터를 추출할 수 있다. 미디어 파일은 /sdcard/NAVER_BAND/ 경로에 캐시로 존재하며, 메시지는 평문으로 저장된다. 로그아웃 시 대부분의 데이터가 삭제되지만, 일부 삭제 메시지는 DB의 상태 플래그(is_deleted=1)를 통해 식별할 수 있다. access_token은 SharedPreferences에 암호화되지 않은 상태로 저장되어 세션 하이재킹의 위험이 존재한다. 2019년 이후 버전에서는 앱 보안이 일부 강화되었으나, 루팅 탐지나 난독화 수준이 낮아 리버스 엔지니어링이 비교적 용이하다. 포렌식 분석 시 삭제 메시지, 파일 업로드 이력, 사용자 활동 로그를 비교적 완전하게 복구할 수 있다.

5.8 Slack

Slack은 2018년 공식적으로 E2EE를 구현하지 않을 것이라고 선언했으며, 기본적으로 전송 중 및 저장 시 암호화만 제공한다. 대신 EKM(Enterprise Key Management)를 제공하여, AWS KMS에 저장된 고객 소유 키로 메시지와 파일을 암호화할 수 있도록 한다. 그러나 EKM도 E2EE가 아닌 서버 측 암호화 강화 방식이므로, 본질적으로 Slack 서버는 메시지 내용에 접근 가능하다. Android 환경에서는 WebView 기반으로 /data/data/com.slack/app_webview/ 경로의 LevelDB에 사용자와 메시지 정보가 저장된다[48]. 로그인 이후 아티팩트가 생성되며, DB Browser for SQLite나 strings 명령어로 일부 사용자 ID와 메시지 내용을 평문으로 추출할 수 있다. Windows에서는 %App-Data%\Slack\ 경로의 LevelDB와 캐시에 메시지, 파일 로그가 존재하며, 삭제 메시지는 거의 복구되지 않는다. Linux에서는 메모리 덤프에서 사용자, 채널, 메시지 등이 JSON 구조로 나타나며, 설계상 민감 정보는 메모리에 저장되지 않도록 되어 있다[61]. Slack은 Electron 기반으로 구축되어 있으며, Chromium 캐시 구조를 따른다. IndexedDB와 LocalStorage에 세션 토큰이 잔존할 가능성이 있다. E2EE가 적용되지 않아 평문 아티팩트 복구 가능성이 높다.

5.9 Signal

Signal은 항상 E2EE가 기본으로 적용되는 메신저로, 자체 개발한 Signal Protocol을 사용하여 모든 메시지와 통화를 암호화한다. 메시지 히스토리는 사용자의 기기에만 저장되며, Signal 서버는 메시지 내용에 접근할 수 없는 제로 지식(zero-knowledge) 아키텍처를 구현하고 있다. Android 환경에서는 /d

ata/data/org.thoughtcrime.securesms/databases/signal.db에 AES 암호화가 적용된 SQLite DB가 사용되며, 사용자 프로필 이미지와 같은 일부 정보는 별도로 저장된다[62]. 복호화를 위해서는 사용자가 설정한 PIN과 Android Keystore에 저장된 키가 필요하며, sqlcipher 명령어나 커스텀 스크립트를 통해 키를 추출한 후 DB를 복호화할 수 있다. 삭제된 메시지는 WAL 파일이나 메모리에 일부 잔류할 수 있으나, Signal의 "disappearing me-ssages" 기능이 활성화된 경우 자동 삭제가 적용된다. 2020년 이후 버전부터는 보안이 더욱 강화되어 스크린샷 방지, 키보드 캐시 비활성화 등의 기능이 추가되었으며, 2024년 3월에는 사용자가 전화번호를 숨기고 사용자명으로 통신할 수 있는 기능이 도입되었다. 전반적으로 E2EE가 강하게 적용되어 포렌식 분석 난이도가 매우 높다.

5.10 Telegram

Telegram은 기본 Cloud 채팅에서는 E2EE를 지원하지 않으며, "Secret Chat"이라는 선택적 E2-EE 기능만 제공한다. Secret Chat은 사용자가 수동으로 활성화해야 하며 1:1 대화에만 사용 가능하고, 그룹 채팅에서는 지원되지 않는다. Android 환경에서는 /data/data/org.telegram.messeng-er/ 경로의 cache4.db, userconfig.xml을 통해 일반 채팅 복원이 가능하며, 표준 포렌식 도구로 D-B 파싱이 가능하다[51,65]. Secret Chat은 E2E-E가 적용되어 로컬에 저장되지 않으므로 분석이 불가능하다. Windows에서는 %AppData%\Telegr-am Desktop\tdata 폴더에 암호화된 세션 정보가 저장되며, 실행 중에만 메모리에서 메시지를 추출할 수 있다[64]. 앱 종료 시 대부분의 데이터가 삭제된다. 웹 환경에서는 IndexedDB와 LocalStor-age에 인증 정보만 저장되고 메시지 본문은 저장되지 않는다[51,63]. Telegram은 MTProto 2.0 프로토콜을 사용하며, 최근 버전에서는 캐시 정리 정책이 강화되었다. 일반 Cloud 채팅은 E2EE 미적용으로 포렌식 분석 난이도가 낮으나, Secret Chat은 분석이 불가능하다.

5.11 Threema

Threema는 기본적으로 E2EE가 적용되는 메신저로, NaCl 오픈소스 라이브러리 기반의 256-bit ECC 암호화를 사용하며, 모든 암호화가 클라이언트 측에서 수행되어 서버를 포함한 제3자가 메시지를 복호화할 수 없다. Android 환경에서는 AES-256으로 암호화된 SQLite DB에 메시지가 저장되며, 복호화를 위해서는 Android Keystore에 보호된 개인 키가 필요하여 추출이 매우 어렵다[62,66]. 중앙 서버 없이 클라이언트 간 키 교환 방식을 사용하며, 스크린샷 차단 등의 보안 기능을 제공한다. 2022년 10월 ETH Zurich 연구팀이 프로토콜 취약점을 발견했으나, Threema는 같은 해 11월 새로운 "Ibex" 프로토콜을 출시하여 해결했다. 기존 연구[62,66]는 Ibex 도입 이전 버전을 대상으로 하므로, 현행 버전의 보안 특성은 해당 연구 결과와 일부 차이가 있다. 강력한 E2EE와 클라이언트 측 키 관리로 인해 포렌식 분석 난이도가 상당히 높다.

5.12 WeChat

WeChat은 E2EE를 지원하지 않으며, 산업 표준 TLS 대신 Tencent가 개발한 커스텀 MMTLS 프로토콜을 사용하여 전송 중 암호화만 제공한다. 중국 본사에 위치하여 중국 정부의 사용자 데이터 요청에 응해야 하는 법적 의무가 있다. Android와 Win-dows 모두에서 SQLCipher로 암호화된 DB를 사용하며, Android에서는 /data/data/com.tenc-ent.mm/MicroMsg/의 EnMicroMsg.db, Win-dows에서는 Documents\WeChat Files\의 M-SG.db에 메시지가 저장된다[67]. SQLCipher D-B 복호화를 위해서는 IMEI와 UIN을 조합한 키가 필요하지만, 해당 키 생성 알고리즘이 공개되어 있어 Python 스크립트나 커스텀 도구를 이용해 복호화를 수행할 수 있다. 삭제 시 secure-delete가 적용되지만, WAL 파일이나 임시 캐시에서 일부 데이터가 남아 복구될 여지가 있다. 미디어 파일은 FileStor-age에 별도 저장되며, 일부 로그는 잔존할 수 있다. 이후 DB 암호화 방식이 강화되었으나, E2EE가 적용되지 않아 Tencent 서버에서 메시지 내용에 접근 가능하다.

5.13 WhatsApp

WhatsApp은 Signal Protocol을 사용하여 모든 메시지, 통화, 공유 미디어에 기본적으로 E2EE

를 적용하며, 2016년 4월 Open Whisper Syste-ms와 협력하여 전체 통신에 대한 E2EE를 완료했다. Windows 환경에서는 %AppData%\Roamin-g\WhatsApp\LocalState\ 경로에 SEE(SQLit-e Encryption Extension)로 암호화된 7개의 D-B와 nondb_settings*.dat 파일이 저장되며, 복호화 키가 nondb_settings*.dat에 이중 암호화되어 저장되어 있으나 커스텀 도구를 통한 키 추출이 가능하다. 웹 환경에서는 IndexedDB와 LocalSto-rage에 AES-128-CBC로 암호화된 메시지가 저장되나, 복호화를 위해서는 서버로부터 salt를 받아야 하므로 오프라인 접근이 어렵다[68]. 전송 중 E-2EE가 적용되어 WhatsApp 서버도 메시지 내용에 접근할 수 없다. 2021년부터 E2EE 백업 기능이 선택적으로 제공되며, 사용자가 활성화하지 않으면 백업은 암호화되지 않는다. 강력한 E2EE와 로컬 암호화가 적용되나 키 추출 가능성과 백업 취약점으로 인해 포렌식 분석 난이도는 높은 수준에 해당한다.

5.14 Wickr

Wickr은 모든 메시지, 통화와 파일 공유에 대해 256-bit E2EE를 기본으로 적용하며, 제로 트러스트 아키텍처를 통해 AWS조차도 메시지 내용에 접근할 수 없도록 설계되었다. 메시지별로 고유한 AE-S 키를 생성하고, Diffie-Hellman 타원곡선 키 교환(ECDH521)을 통해 수신자와 키를 교환한다. Android 및 Windows에서 SQLCipher와 AES-GCM 기반 이중 암호화를 적용한다. Android에서는 .db 파일 외에도 .wic 키 파일, 암호화된 미디어, 평문 로그 등이 존재하며, 자동 로그인을 위해 d-evice ID 기반 키 관리가 수행된다[62]. Windo-ws도 유사한 구조를 갖고 있으며, SQLCipher 도구와 커스텀 Python 스크립트를 사용하여 DB를 복호화할 수 있으나, 키 추출이 매우 어렵다[69]. 전송 중 E2EE가 기본 적용된다. 또한 2021년 AW-S 인수 이후 소비자용 Wickr Me는 2023년 12월 종료되어 서비스 초점이 기업 및 정부로 수렴하였다.

5.15 Wire

Wire는 모든 메시지, 컨퍼런스 콜, 파일에 대해 E2EE를 기본으로 적용하며, Signal Protocol을 기반으로 독립적으로 구현한 Proteus 프로토콜을 사용한다. Proteus는 Curve25519, ChaCha20, HMAC-SHA256 알고리즘을 사용하며, 음성 및 영상 통화는 PFS(Perfect Forward Secrecy)를 지원하는 WebRTC로 암호화된다. Windows 클라이언트는 %AppData%\Wire\ 하위에 Cookies(SQ-Lite), IndexedDB/LevelDB, 로그 등 로컬 아티팩트를 생성하며, 표준 도구(DB Browser for S-QLite, Electron 포렌식 도구)로 메타데이터와 세션 관련 아티팩트를 추출할 수 있다. Cookies에 저장된 세션/크리덴셜 아티팩트로 등록 디바이스 환경에서 로그인 재현이 가능하다[70]. 2017년 Kud-elski Security와 X41 D-Sec의 공동 보안 감사에서 일부 경미 이슈가 확인되었고 수정되었다. 결론적으로 Wire는 콘텐츠 E2EE를 전제하되, 로컬 세션 아티팩트 관리와 로그 처리가 포렌식, 보안 측면의 핵심 쟁점이다.

5.16 종합 비교 분석

본 평가는 디바이스 확보 후 서버 협조 없이 로컬, 클라이언트 아티팩트만으로 사건을 얼마나 재구성할 수 있는지에 초점을 둔다. 복구 난이도는 (1) E2EE 적용 범위(기본/선택/미지원), (2) 로컬 저장 및 키 관리(SQLCipher, DPAPI, 키체인, 세션 및 토큰 잔존), (3) 플랫폼 제약(iOS 샌드박스, 키저장소, 웹 세션 휘발성)의 조합으로 결정된다. Si-gnal, Threema, Wickr는 전면 E2EE 기본과 강한 로컬 보호로 최상 난이도에 위치한다. Whats-App, Element, Wire는 E2EE 기본이지만 백업, 세션 아티팩트, 메타데이터에서 제한적 단서를 제공하므로 상 난이도로 본다. Telegram, Kakao-Talk은 선택적 E2EE(비밀 대화)에 기반하므로 일반 채팅에서 중 난이도로 분류한다. Facebook Messenger는 1:1 E2EE 기본이며 그룹, 일부 기능에 예외가 남아 중간급에 둔다. LINE은 Letter Sealing(E2EE) 기본이되 기능과 버전에 따라 예외가 존재하며, 콘텐츠 포렌식은 어렵지만 일부 아티팩트 분석 여지가 있다. Slack은 EKM 등 관리형 암호화를 제공하지만 진정한 E2EE를 제공하지 않으므로, Discord와 함께 저 난이도에 가깝다. 본 논문에서 제시한 세 가지 복구 난이도 기준으로 각 메신저의 포렌식 증거 수집 가능성을 평가하였다. 이는 현 시점의 기본 설정을 기준으로 한 것으로, 분석 우선순위 결정에 활용 가능하다.

VI. 논의 및 향후 과제

메신저 포렌식 연구는 대체로 탈옥, 루팅, 로그인 유지와 같은 높은 권한이 요구되는 실험 환경에 의존하기 때문에, 실제 수사 현장에 적용할 때 일반화 가능성과 실용성에 제약이 발생한다. 특히 iOS와 웹 클라이언트의 경우, Desktop 및 Android 환경에 비해 연구가 현저히 부족하여 플랫폼별 저장 구조와 복호화 기법이 명확히 정립되지 않은 상태다. 웹 기반 메신저는 브라우저 동작 방식과 세션 관리에 따라 아티팩트의 잔존 여부가 크게 달라지고, 세션 종료 시 대부분의 데이터가 소멸되므로 실시간 수집 전략이 필수적이다. 또한 대부분의 주요 서비스가 Android Keystore나 iOS Keychain 같은 보안 영역에 복호화 키를 저장함에 따라, 데이터 파일을 확보하더라도 해석 자체가 불가능한 사례가 많다.

플랫폼 간 이질성으로 인해 메신저 포렌식 분석 대상과 절차가 일관되지 않는다. 동일한 메신저라도 Windows에서는 파일 시스템에, 웹에서는 IndexedDB에, 모바일에서는 암호화된 데이터베이스에 각각 다른 형태로 저장되며, 플랫폼별로 접근 권한과 추출 방법이 상이하다. 또한 잦은 업데이트로 인해 기존 분석 방법이 무효화되는 경우가 빈번하여, 표준화된 분석 프로세스 정립이 어렵다.

따라서 다음의 후속 연구가 필요하다. 첫째, 권한 제한 환경에서도 동작 가능한 비침투적 수집 기법 개발, 둘째, 본 연구에서 정리한 아티팩트 특성 기반의 자동 인식 및 분석 통합 도구 개발, 셋째, 서비스 업데이트에 대응하는 적응형 포렌식 프레임워크 연구를 진행해야 한다.

VII. 결 론

본 연구는 전 세계 및 국내에서 널리 사용되는 15종 메신저를 대상으로 지난 10년간의 포렌식 연구를 조사하고, 각 서비스의 저장 구조, 암호화 방식, 아티팩트 특성을 비교하였다. 그 결과 Desktop과 Android 환경에서는 증거 획득 기법 연구가 활발히 이루어진 반면, iOS와 웹 기반의 메신저 대상은 상대적으로 연구가 미흡함을 확인했다. 이런 체계적 조사를 통해 플랫폼별, 서비스별 아티팩트 특성을 종합적으로 정리하고, 연구 공백 영역을 명확히 식별하며, 일관된 분석 프레임워크를 제시했다.

이를 기반으로 환경별로 최적화된 분석 절차의 표준화, 낮은 권한 환경에서도 동작 가능한 수집 전략 개발, 그리고 파편화된 연구 결과를 통합하는 자동화 워크플로우 설계가 가능하다. 본 연구가 메신저 포렌식 분야 내 일관된 분석 절차 수립과 자동화 도구 설계의 필요성을 도출하고, 다양한 플랫폼과 조건에 적용 가능한 전략 수립에 기여할 수 있기를 기대한다.

References

[1] J. Yiye, R. Mustapha, M.A. Ahmad, and H. Bassi, "Digital Forensic Investigation of Cyberstalking and Social Media Harassment using Network Forensic Analysis," Journal of Science Technology and Education, vol. 10, no. 3, pp. 1-10, Sep. 2022.

[2] T. Garkava, A. Moneva, and E.R. Leukfeldt, "Stolen data markets on Telegram: a crime script analysis and situational crime prevention measures," Trends in Organized Crime, Online First, Apr. 2024, doi:10.1007/s12117-024-09532-6.

[3] U.S. Drug Enforcement Administration, "National drug threat assessment 2024,", U.S. Department of Justice, May. 2024.

[4] U. Mehta, S. Chougule, R. Mulla, V. Alone, V. K. Borate, and Y. K. Mali, "Instant messenger forensic system," 2024 15th International Conference on Computing Communication and Networking Technologies (ICCCNT), pp. 1-6, Jun. 2024.

[5] D. Barradas, T. Brito, D. Duarte, N. Santos, and L. Rodrigues, "Forensic Analysis of Communication Records of Web-based Messaging Applications from Physical Memory," Proceedings of the 14th International Conference on Security and Cryptography, pp. 43-54, Jan. 2017.

[6] Discord, "Discord," https://discord.com, Accessed: Dec. 04, 2025.

[7] Element, "Element," https://element.io, Accessed: Dec. 04, 2025.
[8] Meta Platforms, "Facebook," https://www.facebook.com, Accessed: Dec. 04, 2025.
[9] Kakao, "KakaoTalk," https://www.kakaocorp.com/page, Accessed: Dec. 04, 2025.
[10] LINE Corporation, "LINE," https://www.line.me, Accessed: Dec. 04, 2025.
[11] Microsoft, "Microsoft Teams," https://www.microsoft.com/ko-kr/microsoft-teams/group-chat-software, Accessed: Dec. 04, 2025.
[12] Naver Corporation, "Naver Band," https://www.band.us, Accessed: Dec. 04, 2025.
[13] Slack Technologies, "Slack," https://slack.com/intl, Accessed: Dec. 04, 2025.
[14] Signal Foundation, "Signal," https://signal.org, Accessed: Dec. 04, 2025.
[15] Telegram, "Telegram," https://telegram.org, Accessed: Dec. 04, 2025.
[16] Threema, "Threema," https://threema.com, Accessed: Dec. 04, 2025.
[17] Tencent, "WeChat," https://www.wechat.com, Accessed: Dec. 04, 2025.
[18] WhatsApp LLC, "WhatsApp," https://www.whatsapp.com, Accessed: Dec. 04, 2025.
[19] Amazon Web Services, "Wickr," https://aws.amazon.com/ko/wickr, Accessed: Dec. 04, 2025.
[20] Wire Swiss GmbH, "Wire," https://wire.com, Accessed: Dec. 04, 2025.
[21] Belkasoft, "Belkasoft Live RAM Capturer," Belkasoft, https://belkasoft.com/ram-capturer, Accessed: Dec. 04, 2025.
[22] Exterro, "FTK Imager," https://www.exterro.com/digital-forensics-software/ftk-imager, Accessed: Dec. 04, 2025.
[23] Velocidex, "WinPmem," https://github.com/Velocidex/WinPmem, Accessed: Dec. 04, 2025.
[24] Basis Technology, "Autopsy," https://www.autopsy.com, Accessed: Dec. 04, 2025.
[25] Magnet Forensics, "Magnet AXIOM," https://www.magnetforensics.com/products/magnet-axiom, Accessed: Dec. 04, 2025.
[26] Paraben Corporation, "Paraben E3 Forensic Platform," https://paraben.com/paraben-e3-forensic-platform, Accessed: Dec. 04, 2025.
[27] X-Ways Software Technology AG, "X-Ways Forensics," https://www.x-ways.net/forensics/index-m.html, Accessed: Dec. 04, 2025.
[28] Oxygen Forensics, "Oxygen Forensic Detective," https://www.oxygenforensics.com, Accessed: Dec. 04, 2025.
[29] Cellebrite, "UFED," https://cellebrite.com/en/ufed, Accessed: Dec. 04, 2025.
[30] MSAB, "XRY," https://www.msab.com/product/xry-extract, Accessed: Dec. 04, 2025.
[31] SleuthKit, "aLEAPP," https://sleuthkit.org/autopsy/docs/user-docs/4.22.0/aleapp_page.html, Accessed: Dec. 04, 2025.
[32] LetsDefend, "How to Install Andriller on Linux," https://letsdefend.io/blog/how-to-install-andriller-on-linux, Accessed: Dec. 04, 2025.
[33] Google, "Android Debug Bridge (adb)," https://developer.android.com/tools/adb, Accessed: Dec. 04, 2025.
[34] Cellebrite, "Physical Analyzer," https://cellebrite.com/en/physical-analyzer, Accessed: Dec. 04, 2025.
[35] MSAB, "XAMN," https://www.msab.com/product/analyze, Accessed: Dec. 04, 2025.
[36] MSAB, "XEC Director," https://www.msab.com/product/manage, Accessed: Dec. 04, 2025.

[37] MSAB, "Kiosk," https://www.msab.com/products/platforms/#kiosk, Accessed: Dec. 04, 2025.
[38] Magnet Forensics, "Magnet Acquire," https://www.magnetforensics.com/resources/magnet-acquire, Accessed: Dec. 04, 2025.
[39] Bring2Lite, "Bring2Lite," https://github.com/bring2lite/bring2lite, Accessed: Dec. 04, 2025.
[40] P. Pawlaszczyk, "FQLite," https://www.staff.hs-mittweida.de/~pawlaszc/fqlite, Accessed: Dec. 04, 2025.
[41] E. Zimmerman, "SQLECmd," https://github.com/EricZimmerman/SQLECmd, Accessed: Dec. 04, 2025.
[42] DB Browser, "DB Browser for SQLite," https://sqlitebrowser.org, Accessed: Dec. 04, 2025.
[43] Exterro, "Forensic Toolkit," https://www.exterro.com/digital-forensics-software/forensic-toolkit, Accessed: Dec. 04, 2025.
[44] KS Tools, "KS DB Merge Tools," https://ksdbmerge.tools, Accessed: Dec. 04, 2025.
[45] Foxton Forensics, "Browser History Capturer," https://www.foxtonforensics.com/browser-history-capturer, Accessed: Dec. 04, 2025.
[46] Foxton Forensics, "Browser History Examiner," https://www.foxtonforensics.com/browser-history-examiner, Accessed: Dec. 04, 2025.
[47] Foxton Forensics, "Browser History Viewer," https://www.foxtonforensics.com/browser-history-viewer, Accessed: Dec. 04, 2025.
[48] Sumin Shin, Eunhu Park, Soram Kim, and Jongsung Kim, "Artifacts analysis of slack and discord messenger in digital forensic," Journal of Digital Contents Society, 21(4), pp. 799-809, Apr. 2020.
[49] K. Gupta, C. Varol, and B. Zhou, "Digital forensic analysis of discord on google chrome," Forensic Science International: Digital Investigation, vol. 44, 301479, Mar. 2023.
[50] Jae-min Cho, Hyeonsu Byun, Hui-seo Yun, Seung-hee Seo, and Changhoon Lee, "Forensic Analysis of Element Instant Messenger Artifacts," Journal of The Korea Institute of Information Security and Cryptology, 32(6), pp. 1113-1120, Dec. 2022.
[51] M.N. Yusoff, A. Dehghantanha, and R. Mahmod, "Forensic investigation of social media and instant messaging services in Firefox OS: Facebook, Twitter, Google+, Telegram, OpenWapp, and Line as case studies," Contemporary Digital Forensic Investigations of Cloud and Mobile Applications, Syngress, pp. 41-62, 2017.
[52] C.S.T. Puri and I. Riadi, "Browser forensic for cyber fraud case on Facebook Messenger services using National Institute of Standard Technology method," International Journal of Computer Applications, vol. 183, no. 42, pp. 22-29, Dec. 2021.
[53] Suhardjono, A.S. Putra, N. Aisyah, and V.H. Valentino, "Analysis of NIST methods on Facebook Messenger for forensic evidence," Journal of Innovation Research and Knowledge, vol. 1, no. 8, pp. 695-702, Jan. 2022.
[54] Sunardi, Herman, and S.R. Ardiningtias, "A comparative analysis of digital forensic investigation tools on Facebook Messenger applications," Journal of Cyber Security and Mobility, vol. 11, no. 5, pp. 655-672, Dec. 2022.
[55] Beomjun Park and Sangjin Lee, "Decryption of KakaoTalk Database for macOS," Journal of The Korea Institute of Information Security and Cryptology,

33(5), pp. 753-760, Oct. 2023.
[56] Minuook Jo and Namsu Chang, "Study on The Data Decryption and Artifacts Analysis of KakaoTalkin Windows Environment," Journal of The Korea Institute of Information Security and Cryptology, 33(1), pp. 51-61, Feb. 2023.
[57] I. Riadi, A. Fadlil, and A. Fauzan, "A Study of Mobile Forensic Tools Evaluation on Android-Based LINE Messenger," International Journal of Advanced Computer Science and Applications (IJACSA), vol. 9, no. 10, pp. 201-206, 2018.
[58] Young-hoon Kim and Taekyoung Kwon, "On Artifact Analysis for User Behaviors in Collaboration Tools - Using differential forensics for distinct operating environments," Journal of The Korea Institute of Information Security and Cryptology, 31(3), pp. 353-363, Jun. 2021.
[59] H. Bowling, K. Seigfried-Spellar, U. Karabiyik, and M. Rogers, "We are meeting on Microsoft Teams: Forensic analysis in Windows, Android, and iOS operating systems," Journal of Forensic Sciences, vol. 68, no. 2, pp. 434-460, Feb. 2023.
[60] Wonseok An and Myungseo Park, "A Study on the Collection and Analysis of Naver Band User Behavior from a Digital Forensics Perspective," Journal of the Korea Institute of Information Security & Cryptology, 34(6), pp. 1263-1272, Dec. 2024.
[61] M. Davis, B. McInnes, and I. Ahmed, "Forensic investigation of instant messaging services on linux OS: Discord and Slack as case studies," Forensic Science International: Digital Investigation, vol. 42, 301401, Jul. 2022.
[62] Jihun Son, Yeongwoong Kim, Dongbin Oh, and Kyounggon Kim, "Forensic analysis of instant messengers: Decrypt Signal, Wickr, and Threema," Forensic Science International: Digital Investigation, vol. 40, 301347, Mar. 2022.
[63] A. Raza, M. Hussain, H. Tahir, M. Zeeshan, M.A. Raja, and Kihyun Jung, "Forensic analysis of web browsers life cycle: A case study," Journal of Information Security and Applications, vol. 85, 103839, Sep. 2024.
[64] P. Fernández-Álvarez and R.J. Rodríguez, "Extraction and analysis of retrievable memory artifacts from Windows Telegram Desktop application," Forensic Science International: Digital Investigation, vol. 40, 301342, Apr. 2022.
[65] C. Anglano, M. Canonico, and M. Guazzone, "Forensic analysis of Telegram Messenger on Android smartphones," Digital Investigation, vol. 23, pp. 31-49, Dec. 2017.
[66] P. Rösler, C. Mainka, and J. Schwenk, "More is Less: On the End-to-End Security of Group Chats in Signal, WhatsApp, and Threema," Proceedings of the 2018 IEEE European Symposium on Security and Privacy (EuroS&P), pp. 415-429, Apr. 2018.
[67] Uk Hur, Myungseo Park, and Jongsung 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 and Cryptology, 30(3), pp. 405-415, Jun. 2020.
[68] Giyoon Kim, Uk Hur, Soojin Kang, and Jongsung Kim, "Analyzing the Web and UWP versions of WhatsApp for digital forensics," Forensic Science International: Digital Investigation, vol. 52, 301861, Mar. 2025.
[69] Sueun Jung, Byeongchan Jeong, Sangj

in Lee, and Jungheum Park, "Forensic Analysis of Wickr Messenger on Windows," Journal of Digital Forensics, 16(4), pp. 42-55, Oct. 2022.
[70] Sumin Shin, Soram Kim, Byungchul Youn, and Jongsung Kim, "Acquiring Credential and Analyzing Artifacts of Wire Messenger on Windows," Journal of The Korea Institute of Information Security and Cryptology, 31(1), pp. 61-71, Feb. 2021.

〈저자소개〉

이 수 연 (Suyeon Lee) 학생회원
2024년 2월 : 성신여자대학교 융합보안공학과 학사
2024년 3월~현재 : 성균관대학교 소프트웨어학과 석사과정
〈관심분야〉 디지털 포렌식, 이상 탐지 모델, 정보보호

샤 흐 저 드 (Shakhzod Yuldoshkhujaev) 학생회원
2024년 8월 : 성균관대학교 소프트웨어학과 학사
2024년 9월~현재 : 성균관대학교 소프트웨어학과 석사과정
〈관심분야〉 시스템 보안, 디지털 포렌식, 정보보호

구 형 준 (Hyungjoon Koo) 종신회원
2010년 2월 : 고려대학교 정보보호학과 석사
2019년 5월 : 스토니 브룩 뉴욕주립대 컴퓨터과학과 박사
2020년 12월 : 조지아텍 박사 후 연구원
2021년 3월~현재 : 성균관대학교 소프트웨어학과 부교수
〈관심분야〉 소프트웨어/시스템 보안, 인공지능을 활용한 정보보안