석사학위논문

클라우드 기반 플랫폼에서의 데이터수집 및 분석기법 연구

2026년

한 성 대 학 교 대 학 원

융 합 보 안 학 과

융 합 보 안 전 공

위 다 빈

석사학위논문
지도교수 박명서

클라우드 기반 플랫폼에서의 데이터수집 및 분석기법 연구

A Study on data collection and analysis techniques on cloud-based platforms

2025년 12월 일

한 성 대 학 교 대 학 원

융 합 보 안 학 과

융 합 보 안 전 공

위 다 빈

석사학위논문
지도교수 박명서

클라우드 기반 플랫폼에서의 데이터수집 및 분석기법 연구

A Study on data collection and analysis techniques on cloud-based platforms

위 논문을 공학 석사학위 논문으로 제출함

2025년 12월 일

한 성 대 학 교 대 학 원

융 합 보 안 학 과

융 합 보 안 전 공

위 다 빈

위다빈의 공학 석사학위 논문을 인준함

2025년 12월 일

심사위원장 석병진 (인)

심 사 위 원 서화정 (인)

심 사 위 원 박명서 (인)

국 문 초 록

클라우드 기반 플랫폼에서의 데이터 수집 및 분석기법 연구

한 성 대 학 교 대 학 원
융 합 보 안 학 과
융 합 보 안 전 공
위 다 빈

본 연구는 클라우드 서비스를 설치형과 비 설치형으로 구분하고, 각 유형에 적합한 데이터 수집 및 분석 절차를 제시하였다. 설치형 서비스인 잔디는 로컬 아티팩트가 제한적으로 잔존하는 특성을 보이므로, 데이터베이스와 캐시 등 로컬 아티팩트를 우선적으로 분석한 뒤 주요 API를 식별하고, 메모리에서 확보한 인증 정보를 활용해 API 요청을 재구성하여 서버 데이터를 직접 획득하는 절차를 도출하였다. 이후 가상 시나리오를 구성하여 해당 절차를 적용하고, 실제 채팅방을 재구성함으로써 분석 과정의 유효성을 검증하였다. 비 설치형 서비스인 네이버톡·디스코드·폴라리스 오피스는 웹 브라우저 기반으로 동작하여 로컬에 의미 있는 데이터가 거의 남지 않기 때문에, 본 연구에서는 먼저 크리덴셜을 확보하고 이를 이용해 API 호출을 수행함으로써 서버 데이터를 수집한 뒤, 수집된 데이터를 분석하였다.또한 메시지 삭제, 숨기기, 파일 삭제 등 안티 포렌식 기능에 따른 데이터 변화를 비교하여 서비스별 데이터 잔존성을 검토했다. 마지

막으로 각 서비스별 데이터 수집 도구를 제작하여 실제 환경에서 절차의 유효성을 검증하였다. 본 연구의 결과는 클라우드 기반 협업 도구 및 메신저에 대한 디지털 포렌식 절차 수립에 참고 가능한 예시를 제공해, 증가하는 클라우드 환경 수사에서 표준화된 분석 방법론 마련에 기여할 것으로 기대한다.

【주요어】Digital Forensics, Cloud Forensics, JANDI, Naver Talk, Discord, Polaris Office

목 차

표 목 차

그 림 목 차

제 1 장 서론

제 1 절 연구 배경

기술의 발전과 함께 하드웨어 장비나 소프트웨어를 직접 보유하지 않고 필요한 만큼 빌려 사용하는 클라우드 서비스는 분산 저장, 실시간 동기화, 다중 디바이스 지원 등 다양한 장점을 제공하며 개인·기업·기관의 핵심 인프라로 자리 잡았다. 이에 따라 사용자는 단일 기기에 의존하지 않고 클라우드 서버를 매개로 다양한 서비스를 이용할 수 있게 되었다.

이와 더불어 서비스 형태 역시 지속적으로 변화하였다. 코로나19 확산 이후 비대면 업무 수요가 급격히 증가하면서 클라우드 기반 서비스의 도입이 확대되었고, 프로그램 설치 없이도 웹 브라우저만으로 이용할 수 있는 비 설치형 서비스가 등장하면서 접근성과 편의성도 함께 향상되었다. 데이터가 사용자 기기가 아닌 서버에서 처리·저장되는 구조가 보편화되면서, 디지털 포렌식 관점에서는 로컬 분석 중심의 기존 방식으로 대응하기 어려운 문제가 새롭게 등장했다.

전통적인 포렌식은 로컬 장치에 남는 아티팩트를 중심으로 수행되었으나, 클라우드 환경에서는 로컬 흔적이 제한적이거나 존재하지 않을 수 있다. 또한 클라우드 서비스는 협업도구, 메신저, 오피스 서비스 등 다양한 목적과 구조가 서로 다르기 때문에 서비스별로 데이터 위치, 잔존성, 인증 구조가 상이하다. 이로 인해 단일 접근 방식으로는 포렌식 분석을 수행하기 어렵고, 서비스 유형에 따른 포렌식 분석 체계가 필요함을 의미한다.

본 논문은 이러한 문제 인식을 바탕으로 클라우드 서비스를 설치형과 비 설치형 서비스로 분류하여 각 서비스 유형에 적합한 포렌식 분석 방법을 제시한다.

제 2 절 연구 목적

본 연구의 목적은 클라우드 환경에서 데이터 저장 방식이 변화함에 따라 전통적인 로컬 기반 포렌식이 지니는 한계를 보완하고, 서비스 유형별로 적합한 분석 절차를 제안하는 것이다. 이를 위해 클라우드 서비스를 설치형과 비 설치형으로 구분하고, 각 유형에서 생성되는 데이터의 특성과 접근 가능성을 포렌식 관점에서 체계적으로 검토하고자 한다.

설치형 서비스는 로컬 아티팩트 분석과 API 기반 데이터 재구성을 수행하고, 가상 시나리오를 통해 획득한 데이터를 재구성하는것을 보임으로서의 검증 및 재현 가능성을 확인한다. 비 설치형 서비스는 크리덴셜 기반 데이터 수집과 안티 포렌식 행위 분석을 수행함으로써 클라우드 환경에서 활용 가능한 포렌식 접근 방법을 제안한다.

연구 대상은 설치형 서비스의 대표 사례로 협업도구 잔디를, 비 설치형 서비스로는 메신저형 서비스(네이버톡, 디스코드)와 오피스형 서비스(폴라리스 오피스)를 선정하였다. 각 서비스의 데이터 구조, 인증 방식, API 요청 패턴을 분석하여 클라우드 포렌식 관점에서 데이터 수집의 가능성과 한계를 검증하고, 서비스 유형별 차이를 반영한 포렌식 절차 제안에 기여하고자 한다.

본 연구는 로컬 아티팩트가 제한적으로만 잔존하는 클라우드 기반 서비스 환경에서 기존의 로컬 아티팩트 중심 디지털 포렌식 접근만으로는 충분한 분석이 어렵다는 점을 고려하여, 서버에 저장된 데이터를 수집·분석하기 위한 절차를 정리하고자 하였다. 이러한 서비스 중 설치형과 비 설치형 클라우드 서비스를 구분하고, 각 서비스 유형에서 적용 가능한 데이터 수집 경로와 분석 절차를 정리하였다. 또한 네트워크 통신 분석과 크리덴셜 기반 API 요청 재구성을 결합한 분석 방법을 사례에 적용하여 그 활용 가능성을 검토하였다. 이러한 연구 결과는 클라우드 환경을 대상으로 한 디지털 포렌식 분석 절차를 이해하고 향후 관련 연구를 수행하는 데 참고 자료로 활용될 수 있을 것으로 기대된다.

제 2 장 클라우드 서비스

제 1 절 클라우드 서비스의 개념 및 유형

클라우드 서비스는 네트워크를 통해 원격의 컴퓨팅 자원과 소프트웨어 기능을 제공하는 기술로, 사용자가 장비를 직접 구축하거나 관리하지 않고도 필요한 기능을 즉시 이용할 수 있다는 점에서 기존 온프레미스 환경의 한계를 극복하였다. 클라우드 컴퓨팅은 서버, 저장 장치, 네트워크, 애플리케이션을 중앙화된 환경에서 제공함으로써 초기 구축 비용과 유지·관리 부담을 줄이고, 필요한 만큼 자원을 확장·축소할 수 있는 유연성을 제공한다. 또한 인터넷이 연결된 환경이라면 언제 어디서나 동일한 자원에 접근할 수 있어 업무 효율성과 이동성을 크게 향상시킨다.
클라우드 서비스 제공 방식은 일반적으로 IaaS(Infrastructure as a Service), PaaS(Platform as a Service), SaaS(Software as a Service)의 세 가지 모델로 구분된다.

IaaS는 서버·스토리지·네트워크와 같은 기본 인프라 자원을 가상화하여 제공하는 형태로, 사용자는 필요한 용량만큼 인프라를 구성할 수 있다. PaaS는 인프라 위에 애플리케이션 개발 및 운영에 필요한 플랫폼과 미들웨어를 제공하여 개발 환경 구축을 단순화한다. SaaS는 클라우드에서 호스팅되는 애플리케이션을 사용자가 인터넷을 통해 직접 이용하는 방식으로, 별도 설치나 유지 보수 없이 기능을 제공받을 수 있다는 점에서 가장 널리 활용되는 모델이다.

본 연구의 분석 대상인 잔디, 네이버톡, 디스코드, 폴라리스 오피스는 모두 SaaS형 서비스에 해당한다. 사용자는 애플리케이션 또는 웹 브라우저를 통해 서비스 제공자의 서버에 저장된 데이터와 기능에 접근하며, 서비스 내부 로직과 데이터 저장은 클라우드 인프라에서 처리된다. 이 중 잔디는 데스크톱 클라이언트를 설치하여 일부 데이터를 로컬에 캐싱하는 하이

브리드 구조를 가지는 반면, 네이버톡·디스코드·폴라리스 오피스는 웹 기반 접근을 중심으로 대부분의 데이터를 서버에서 관리하는 비 설치형 구조를 가진다.

본 논문은 클라우드 서비스를 설치형과 비 설치형 서비스로 구분하여 연구를 진행하였다. 설치형과 비 설치형 서비스는 데이터 생성 위치, 로컬 아티팩트 잔존성에서 차이를 보이며, 이러한 구조적 차이는 포렌식 분석 방법에도 영향을 미친다. 이에 따라 각 서비스 유형의 기술적 특성은 본 연구에서 제시하는 클라우드 포렌식 절차 설계의 핵심 기준이 된다.

제 2 절 클라우드 포렌식의 개요

클라우드 포렌식은 클라우드 환경에서 발생한 디지털 증거를 수집하고 분석하는 프로세스를 의미하며, 기존 로컬 기반 디지털 포렌식의 확장 개념이다. 클라우드 서비스는 가상화 기술을 기반으로 다양한 서버와 저장소에 데이터를 분산하여 처리하므로, 데이터가 단일 장치에 저장되던 전통적 환경과 구조적 차이를 가진다. 이러한 특성으로 인해 클라우드에서의 증거 확보 방식은 로컬 장치 중심의 포렌식 방법을 적용하기 어렵다.

클라우드 서비스의 주요 특징은 데이터의 물리적 위치 파악이 어렵고, 사용자 단말에 남는 로컬 아티팩트가 제한적이라는 점이다. 또한 사용자에게 제공되는 서버, 스토리지, 서비스 등이 물리적으로 분리되어 있으며, 세션 기반의 접근이 일반적이기 때문에 크리덴셜 정보가 증거 수집 과정에 중요한 요소로 작용한다.

클라우드 기반 서비스에서는 대부분의 사용자 활동이 API 요청 형태로 이루어지고, 데이터 기록 방식 또한 서비스마다 다르게 구현된다. 따라서 서비스의 유형 파악, API 함수 파악 및 응답 데이터를 정리하는 것이 클라우드 포렌식 절차 설계의 핵심이다. 본 연구에서는 이러한 특성을 고려하여 설치형과 비 설치형 SaaS 서비스를 구분하고, 서비스 유형별로 적용 가능한 포렌식 분석 방법을 제시하고자 한다.

제 3 절 선행 연구

디스코드와 관련된 다양한 디지털 포렌식 분석 연구가 진행되었다. Megan Davis 외 2인은 Linux 환경에서 디스코드와 슬랙의 메모리 포렌식을 수행하여 RAM으로부터 사용자명, 이메일, 비밀번호, 메시지, 첨부파일 등을 복원할 수 있음을 확인하고, 메모리 포렌식이 중요한 증거 수단이 될 수 있음을 보여주었다. Khushi Gupta 외 2인은 Google Chrome 브라우저 환경에서 실행된 디스코드를 대상으로 분석을 수행하여 결제 정보, 메시지, 계정 설정, 첨부파일 등을 복구할 수 있음을 제시하였다. 신수민 외 3인은 슬랙과 디스코드를 분석하여 모바일 애플리케이션과 PC 프로그램에서 주요 아티팩트를 정리하고, 이를 바탕으로 포렌식적으로 활용 가능한 시나리오를 제시하였다. 또한 Khushi Gupta 외 2인은 Windows와 Linux 환경에서 디스코드의 아티팩트를 식별하였으며, 로컬 저장소에서는 메시지, 채널, 서버 등의 데이터를, RAM에서는 계정 정보, 비밀번호, 사용자 친구 정보, 위치 정보, 채팅 및 서버 정보를 확인하였고, 네트워크 패킷 캡처를 통해 디스코드 API와의 통신을 추적하였다. Farkhund Iqbal 외 2인은 Windows, macOS, Linux 환경에서 디스코드 애플리케이션의 로컬 아티팩트를 조사하여 캐시로부터 채팅 로그를 복구하고, Local Storage로부터 계정 정보 및 서버 관련 데이터를 확보하였으며, 이를 기반으로 캐시와 Local Storage에서 정보를 추출할 수 있는 'DiscFor'라는 도구를 개발하였다. Kyle Moffitt 외 3인은 Windows 10 환경에서 디스코드 데스크톱 애플리케이션을 분석하여 타이핑된 메시지 초안의 평문 사본과 캐시된 사진을 복구할 수 있음을 확인하였다.

협업 도구와 관련된 연구도 진행되었는데, 김영훈 외 1인은 Microsoft Teams에 대해 Windows 및 Android 환경에서 주요 사용자 행위를 정의하고 아티팩트 존재 여부를 검증한 뒤 차분 포렌식을 통해 협업 툴 분석 기법 및 수사 시나리오를 제시하였다. 김한결 외 2인은 Windows 환경에서 네이버 웍스를 대상으로 로컬 아티팩트를 식별하고 기능별 데이터를 수집·분류하였으며, 암호화된 데이터베이스를 복호화하고 안티 포렌식 기

능에 따른 아티팩트 변화를 분석하였다. Herschel Bowling 외 3인은 Windows 10, Android, iOS 환경에서 Teams의 아티팩트를 수집·분석하여 수작업 조사를 통해 77.6%를 복구할 수 있었으나, Cellebrite UFED와 Magnet AXIOM Examine 도구를 통한 자동 수집에서는 13.8%만 복구되어 자동화의 한계를 지적하였다.

마지막으로 폴라리스 오피스에 대한 연구에서는 이연주 외 2인이 Windows와 macOS 환경에서 폴라리스 오피스 PC 버전 사용 시 남는 아티팩트를 조사하여 데이터베이스와 plist 파일이 중요한 역할을 하며, 이를 통해 사용자의 작업 과정과 과거 이력을 추적할 수 있음을 제시하였다. 기존 연구들은 주로 설치형 애플리케이션 또는 로컬 기반 협업 도구를 중심으로 진행되어, 디스크 저장소·레지스트리·캐시·메모리 등 로컬 아티팩트의 수집과 복구에 초점이 맞추어졌다. 이에 따라 브라우저 기반으로 동작하는 비 설치형 클라우드 서비스에 대한 분석은 상대적으로 부족하였다. 이에 본 연구는 기존의 로컬 중심 포렌식에서 확장하여 클라우드 포렌식, 즉 비 설치형 클라우드 서비스의 네트워크 통신을 재구성하여 확보한 데이터를 분석 및 분류하고, 각 서비스별 안티 포렌식 행위가 데이터 보존에 미치는 영향을 정리하였다. 마지막으로 웹 기반 클라우드 포렌식의 분석 절차를 제시함으로써, 비 설치형 클라우드 서비스를 수사 과정에서 활용할 수 있는 가능성을 제시하였다.

제 4 절 분석 환경

본 논문에서는 설치형 클라우드 서비스 사례로 Windows 환경에서 동작하는 잔디를 대상으로 로컬 아티팩트 및 데이터 획득 방법을 분석하였다. 분석에 사용된 도구·환경 및 대상 소프트웨어는 Table 1.과 같다.

Table 1. Analysis environment and tools for Installable cloud service

Type	Name	Version
Collaboration Tool	JANDI	1.7.6
Desktop	Windows 10 pro	-
Hex Viewer	HxD	2.5.0.0
Database Viewer	DB Browser for SQLite	3.29
CacheViewer	ChromeCacheView	v2.46
Data Transformer	Chromium_dump_local_storage.py	0.1
Proxy	Fiddler	v5.0.20211.51073
memory acquisition	winPmem	4.0

잔디는 1.7.6 버전을 대상으로 분석하였으며 요금제는 가장 높은 ENTERPRISE 요금제를 사용하였다. 협업 도구 분석은 윈도우 10 pro 환경의 데스크탑에서 진행하였고, 아티팩트 분석을 위해 데이터베이스 파일은 DB Browser for SQLite를 사용하여 확인하였으며 로우 데이터 확인을 위해 HxD를 사용하였다. 또한 캐시 형식의 데이터를 분석하기 위해 ChromeCacheView를 사용하였으며. levelDB 형식의 데이터를 DB 형식으로 변환하기 위하여 cclgroupltd에서 제작한 Chromium_dump_local_storage.py를 사용하였다. 또한 잔디의 API를 분석하기 위하여 프록시 도구인 Fiddler를 사용하였다. 그리고 메모리에서 데이터를 덤프하기 위해 winPmem을 사용하였다.

다음으로 비 설치형 클라우드 서비스는 윈도우 환경에서 웹 브라우저를 이용해 동작하는 인스턴트 메신저인 네이버톡 및 디스코드와 오피스 소프트웨어인 폴라리스 오피스를 대상으로 클라우드 데이터 획득 방법을 분석하였다. 이 과정에서 사용된 분석 도구 및 환경, 분석 대상 협업 도구와 분석에 사용한 소프트웨어는 Table 2.와 같다.

Table 2. Analysis environment and tools for non-installable cloud service

Type	Name	Version
Instant Messenger	Discord	(web)
	Navertalk	
Office Software	Polaris office	
Desktop	Windows 10 pro	-
Web Browser	Chrome	140.0.7339.128
Hex Viewer	HxD	2.5.0.0
Proxy	Fiddler Everywhere	7.2.0
memory acquisition	winPmem	4.0

네이버톡, 디스코드, 그리고 폴라리스 오피스는 모두 웹 버전을 대상으로 분석을 수행하였으며, 웹 브라우저로는 크롬을 사용하였다. 메모리 덤프에는 winPmem을 활용하였고, 덤프 된 메모리로부터 각 서비스의 크리덴셜을 확인하기 위해 HxD를 사용하였다. 마지막으로 클라우드 서비스들의 API를 분석하기 위하여 프록시 도구인 Fiddler Everywhere를 사용하였다.

제 3 장 방법론

제 1 절 주요 API 식별 및 재구성

1) 주요 API 식별

클라우드 기반 서비스는 대부분 클라이언트와 서버 간의 통신을 API 요청·응답 구조로 처리하며, 서버에 저장된 데이터를 획득하기 위해서는 서비스가 사용하는 주요 API를 식별하는 과정이 필수적이다. 따라서 포렌식적으로 의미 있는 데이터를 획득하기 위해서는, 서비스가 실제로 사용하는 API의 종류와 요청 구조를 식별하는 과정이 선행되어야 한다. 이를 위해 공개적으로 확인 가능한 API 엔드포인트 정보와, 프록시 도구를 이용해 캡처한 실제 네트워크 통신 내역을 병행하여 분석하였다. 공개된 정보는 API 구조와 기능을 파악하는 기준으로 활용하였으며, 프록시 도구를 통해 확인된 네트워크 패킷은 실제 서비스 환경에서 사용되는 API 요청과 파라미터 구성을 식별하는 데 활용하였다.

먼저, 공개된 API 문서 참조의 경우 서비스 제공자가 공개한 개발자 문서 또는 공식적으로 공개된 API 엔드포인트를 참조하여 API 의 역할 및 실제로 구동이 가능한지 확인하였다.

다음으로, 프록시를 통한 확인의 경우, 프록시 도구를 이용해 애플리케이션 또는 웹 서비스가 구동 중인 환경에서 네트워크 패킷을 캡처하고, 클라이언트와 클라우드 서버 간에 이루어지는 HTTP 및 웹소켓 통신을 캡쳐하였다. 그리고 캡처된 트래픽을 기반으로 클라우드 서버와 통신하는 주요 API 요청을 식별하고, 각 요청에 포함되는 HTTP 메서드, URL, 헤더, 파라미터, 바디의 내용을 분석하였다.

2) API 요청 재구성

식별된 API를 기반으로, 서버 데이터를 획득하기 위해 API 요청 재구성을 수행하였다. 캡처된 네트워크 패킷을 분석하여 API 요청에 필요한 파라미터와 각 요청의 응답으로 획득되는 값을 구분하고, 인증에 사용되는 값과 특정 자원을 식별하기 위한 파라미터를 정리하였다.

이후 Python3 환경에서 requests 모듈을 사용하여 API 요청을 재구성하였다. 재구성 과정에서는 원본 패킷과 동일한 HTTP 메서드와 헤더 구성을 유지하였으며, 사용자 인증을 위한 크리덴셜과 각 API에서 요구하는 파라미터를 올바르게 설정하였다. 재구성된 API 요청을 통해 서버로부터 반환되는 응답 데이터는 JSON 형식으로 수집되었으며, 이를 분석하여 채팅 내용, 메타데이터, 파일 정보 등을 구조화하였다.

제 2 절 크리덴셜 확보

1) 메모리를 통한 크리덴셜 확보

클라우드 서비스는 사용자가 로그인한 상태에서 세션 유지를 위해 인증 토큰이나 쿠키 값을 메모리 상에 저장한다. 이에 본 연구에서는 분석 대상 서비스가 실행 중인 상태에서 메모리 덤프를 수행하고, 덤프된 메모리 데이터를 분석하여 인증 토큰 및 세션 관련 값을 추출하였다. 이 과정에서 특정 문자열이나 문자열 패턴을 기준으로 크리덴셜을 식별하였다.

2) 웹 브라우저를 통한 크리덴셜 확보

비설치형 클라우드 서비스는 웹 브라우저 기반으로 동작하며, 인증을 위해 쿠키 및 세션 정보를 활용하는 구조를 가진다. 이에 본 연구에서는 웹 브라우저의 개발자 도구를 이용하여 쿠키 및 세션 정보의 위치를 확인하고, 서비스에 대응되는 인증 쿠키 값을 식별하였다. 또한 확보한 크리덴셜이 메모리 덤프 상에서도 동일하게 존재하는지를 확인하였다.

제 4 장 설치형 클라우드 서비스

제 1 절 잔디

1) 애플리케이션 개요

협업 도구는 조직의 업무를 돕기 위해 만들어진 도구이다. 따라서 조직 내에 권한이 존재하듯 조직이 사용하는 사용자 권한을 제공하며 Table 3. 과 같이 사용자를 분류한다.

Table 3. JANDI's permission classification

Authority	Role
Team owner	Team management, content management, transfer of owner rights, deletion of team
Team manager	Team management, content management
Regular member	Access to public topics and content. You own the content you created, use JANDI's general functions
Associate member	Access only to invited topics and shared content

잔디의 권한은 팀 소유자, 팀 관리자, 정회원, 준회원으로 나뉜다. 관리자로 분류되는 팀 소유자와 팀 관리자는 공통으로 팀 관리 및 컨텐츠 관리를 수행할 수 있다. 팀 소유자는 추가적으로 소유자 권한 이양 및 팀 삭제에 대한 권한을 갖는다. 일반 사용자로 분류되는 정회원 및 준회원은 일반 사용자 컨텐츠에 참여할 수 있다는 공통점이 존재하지만, 준회원은 초대된 컨텐츠에만 접근이 가능하다는 제약이 존재한다. 잔디의 권한 중 관리자 권한인 팀 소유자와 팀 관리자는 다른 권한은 사용할 수 없는 관리자 메뉴를 사용할 수 있으며, 그 내용을 정리한 표는 Table 4.와 같다.

Table 4. JANDI's manager function

Function	Explanation
Member Management	Identification of all participating members, member-specific management functions
Team Management	Admin settings for the entire group, change the team name and domain, delete teams, and set whether to allow deletion of 1:1 conversations.
Download history	Check download history by member/period
Additional function settings	Message deletion function activation setting, Message reception confirmation setting

멤버 관리에서는 전체 멤버를 조회할 수 있으며, 멤버별 관리 설정을 할 수 있다. 팀 관리에서는 전체 그룹에 대한 설정 및 1:1 대화 삭제 허용 여부 등 팀에 일괄 적용되는 설정이 가능하다. 잔디의 일반 사용자가 사용할 수 있는 기능은 Table 5.와 같다.

Table 5. JANDI's general user features

Function	Explanation
chatting	1:1 chat, chat with me, group chat
topic	Public/private group chat
drive	Share files through Drive
to do	Schedule creation function
calender	calendar function
video meeting	voice and video calls
vote	Voting function for people in the group
watermark	Watermark function when printing documents

잔디의 메신저 기능은 채팅과 토픽으로 구성되어 있다. 채팅과 토픽의 차이는 대화의 주제 유무, 참여 가능 멤버 수 이다. 토픽은 주제별 대화방이며 공개 토픽일 경우 모든 정회원 멤버가 참여 가능하고, 비공개 토픽일 경우 초대를 받아야만 참여할 수 있다. 또한 비공개 토픽은 다른 멤버에게 공개되지 않는데, 초대받지 않았다면 팀 관리자이더라도 열람하거나 검색할 수 없다. 채팅은 대화의 주제가 존재하지 않는 자유로운 대화방으로 최대 10명과 대화할 수 있으며 준회원과도 대화가 가능하지만, 정회원 멤버의 초대를 통해야만 한다. 또한 팀 관리자이더라도 참여하지 않은 채팅을 열람하거나 검색할 수 없다. 드라이브를 통한 파일 공유 및 캘린더 및 할 일 기능을 통해 일정을 작성할 수 있다. 투표 기능을 통해 그룹 내 인원

간 특정 주제에 대해 투표가 가능하며 워터마크 기능을 통해 잔디를 통해 출력하는 출력물을 대상으로 워터마크가 적용된다. 모든 기능은 웹과 클라이언트에서 동일하게 사용이 가능하다.

2) 잔디의 로컬 아티팩트 수집 및 분석

윈도우 10 OS를 사용 중인 PC의 'C:\Users\[USERNAME]\AppData\Roaming\JANDI' 경로에는 잔디 애플리케이션의 아티팩트가 존재한다. 해당 경로에는 잔디가 동작하는 과정에서 생성되는 여러 종류의 데이터가 저장되어 있었으며, 본 연구에서는 해당 데이터들을 파일 유형별로 세밀하게 분류하였다. 각 파일 유형별 특성에 맞추어 적절한 분석 방법을 통해 데이터를 확인하였고, 이를 통해 잔디의 아티팩트 저장 구조를 확인하였다. 잔디는 클라이언트 설치 후 동작 시 Fig. 1.과 같은 형태로 애플리케이션 데이터가 생성된다.

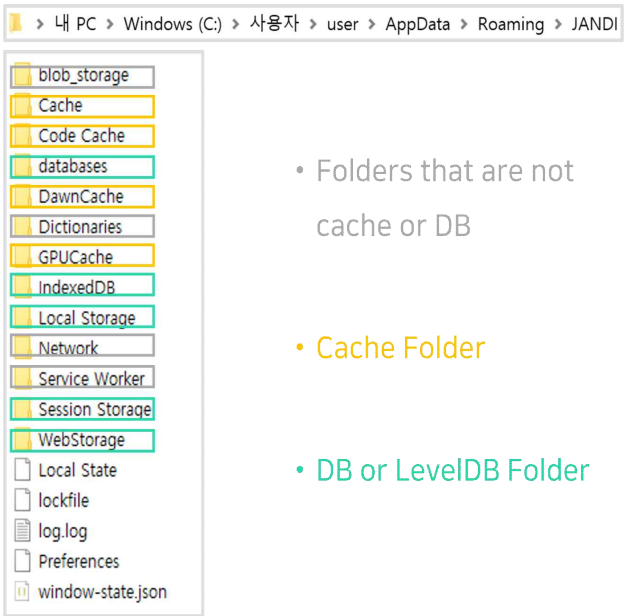

Fig.1. JANDI's local artifact structure

해당 경로에서 획득할 수 있는 디지털 포렌식 관점에서 의미 있는 데이터는 Cache와 Local Storage 하위에 존재한다. 먼저 Cache 폴더를 확인하기 위해서 ChromeCacheView라는 도구를 활용하여 캐시 폴더 내의 정보를 확인하였다. Cache 폴더에는 회사의 로고와 1대1 채팅방, 그룹 채팅방, 잔디에서만 존재하는 그룹 채팅방인 토픽 등 다양한 종류의 채팅방에서 주고받은 사진들의 썸네일이 존재하였으며, 해당 내용에 대해 획득한 썸네일과 실제 잔디 내의 환경 및 채팅방별로 상세하게 정리한 그림은 Fig. 2.와 같다.

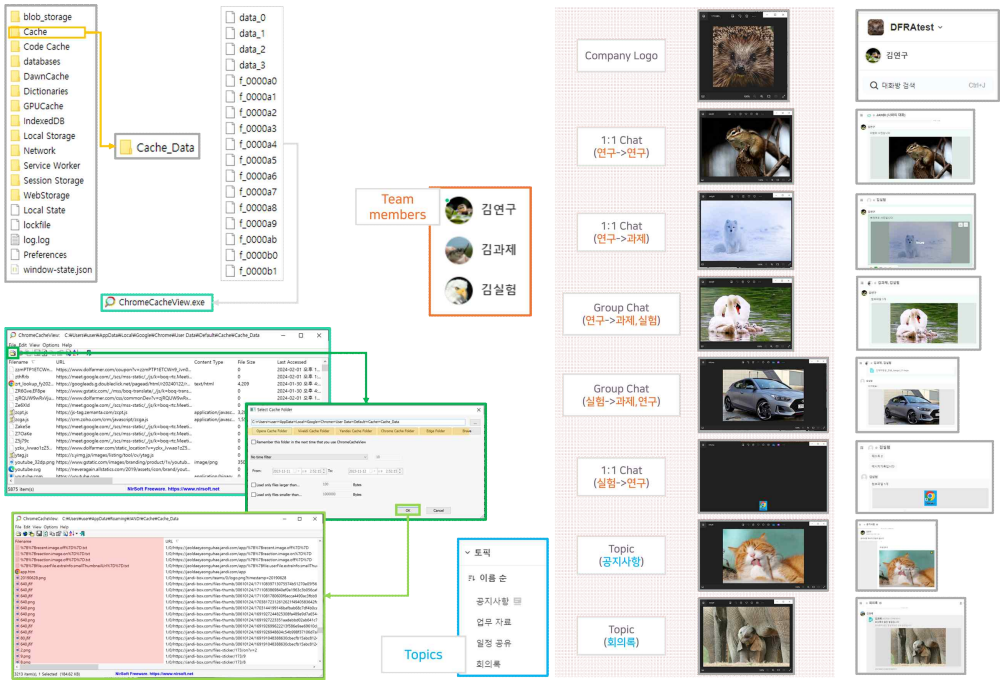

Fig.2. JANDI's Cache Data Identification

사진들의 확장자는 .jfif 또는 png이며 모든 사진은 "640"이라는 동일한 이름으로 존재하였다. 다음으로, Local Storage는 LevelDB 파일 형식으로 저장되어 있으며, 이를 분석하기 위해 오픈소스 파싱 도구인 Chromium_dump_local_storage.py를 활용하여 SQLite 형식의 DB로 추출하였다. 그리고 DB Browser for SQLite 도구를 사용하여 데이터의 내용

을 시각적으로 확인하여 데이터를 식별하였다. 해당 DB에서 중요한 테이블은 "records_view"였으며, 해당 테이블에서 얻을 수 있는 정보는 Table 6.과 같다.

Table 6. Key data within local artifacts

Folder	Cache	Local Storage		
Data obtained	Company logo, Thumbnails of photos exchanged in chat rooms and topics	storage_key	key	value
		https://.[Team domain name].com	_jd_.team_name	Team name of this computer user
			_jd_.team_id	Team ID of this computer user
			_jd_.member_id	Member ID of this computer user

storage_key 컬럼의 팀도메인은 잔디의 관리자 메뉴 속 팀 관리 탭의 팀도메인 변경 항목의 웹 주소와 동일하다. value 컬럼의 내용은 3장의 주요 API 분석을 통해 확인하였다. _jd_.team_name은 잔디에 로그인한 사용자의 팀명을 의미하며, 잔디 프로그램의 상단에 위치한 팀명과 동일하다. _jd_.team_id와 _jd_.member_id는 각각 잔디에 로그인한 사용자가 속해 있는 팀의 고유번호와 사용자의 고유번호를 의미한다. 이중 _jd_.member_id의 값은 API 요청 재구성을 위한 파라미터에 속하지는 않지만, 팀 내 멤버 개인을 구분하는 번호이기 때문에 API 응답 값 중 fromEntity, writerId 등으로 표현되는 고유번호를 식별할 때 송신자를 특정하는 증거로 사용할 수 있다.

3) 주요 API 분석

잔디는 서버 측에서 데이터를 요청하여 클라이언트에서 재구성하는 방법을 사용하기 때문에 로컬에는 서비스를 제공하기 위한 최소한의 데이터

만 존재한다. 따라서, 포렌식적으로 의미 있는 데이터를 획득하기 위해서는 서버에 저장된 데이터에 접근할 필요가 있다. 이를 위해 잔디가 서버 측에 데이터를 요청할 때 사용하는 API를 식별하고 주요 데이터에 접근할 수 있는 API 요청 재구성 방법에 대해 분석하였다. 웹 프록시 도구인 fiddler를 이용하여 분석한 잔디의 주요 API는 Table 7.과 같다.

Table 7. JANDI's main API

ID	URL	Acqusition data	Parameter
teamID search (API-1)	https://i1.jandi.com/inner-api/account	'teamID' where the token user is registered	Token, "Accept": "application/vnd.tosslab.jandi-v3+json"
roomID search (API-2)	https://i1.jandi.com/start-api/v4/teams/{teamID}/rooms	'roomID' of all chat rooms the user is participating in	Token, teamID
Chat search (API-3)	https://i1.jandi.com/message-api/v1/teams/{teamID}/rooms/{roomID}/messages?count={Number of data to import}	Full conversation content, chat time, creator ID, sent/received file name	Token, teamID, roomID
Search file list (API-4)	https://i1.jandi.com/search-api/v1/teams/[teamID]/search?accessType=joined&count=20&fileType=all&order=desc&page=1&roomId=[roomID]&type=file	File name, fileID, target post, author ID, creation time	Token, teamID, roomID, Number to search, file type, page number
Download file (API-5)	https://i1.jandi.com/file-api/v1/teams/[teamID]/files/[fileID]/downloadUrl	File download URL, link expiration time (Unix timestamp)	Token, fileID, teamID

최종적으로 채팅이나 파일을 얻기 위해서는 토큰, teamID, roomID, fileID가 필요하다. 사용자 토큰은 클라우드 서버에 데이터를 요청할 때 인가된 사용자임을 증명할 수 있는 인증 정보이다. teamID, roomID는 잔디의 팀 및 특정 채팅방에 대한 고유한 ID 값이며 fileID는 잔디 내에서 공유된 파일을 식별할 수 있는 고유한 ID 값이다. 이러한 인증 정보 및 고유 ID 값은 API 통신 시 HTTP 요청에 포함되어 인증 절차를 통과하거나, 특정 자원을 지칭하는 용도로 사용된다. 따라서 선행적으로 각 데이터를 획득하고 이를 이용한 올바른 요청을 구성해야 한다. 하지만 이러한 데

이터는 바로 획득할 수 없기 때문에 다른 API와의 유기적인 연결이 필요하다. API-1은 토큰값과 특정 문자열을 이용하여 토큰 사용자의 teamID를 얻을 수 있다. 이렇게 얻은 teamID를 API-2에 토큰과 함께 사용하면 사용자가 참여 중인 모든 채팅방의 roomID를 얻을 수 있다. roomID는 API-3, API-4에서 사용되는 파라미터로 각각의 추가적으로 요구되는 파라미터와 사용 시 API-3로부터는 채팅 내용을 재구성할 수 있는 데이터인 대화 내용, 채팅 시각, 작성자ID, 송/수신한 파일명 등을 얻을 수 있으며, API-4로부터는 API-5에서 요구되는 파라미터인 fileID를 얻을 수 있다. 토큰, fileID, teamID를 이용하여 API-5에 데이터를 요청할 경우 파일을 다운로드할 수 있는 URL을 얻을 수 있다. 위 내용을 직관적으로 구성한 그림은 Fig. 3.과 같다.

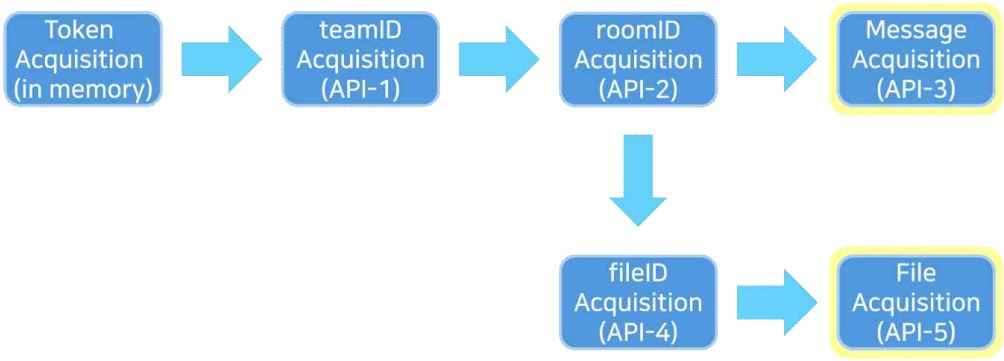

Fig. 3. Data acquisition procedure through API

잔디의 API 중 데이터 조회 및 다운로드에 사용되는 API에 대해 HTTP 요청을 재구성하면 잔디 서버에 있는 데이터를 가지고 올 수 있으며, 이렇게 획득한 데이터는 JSON 또는 파일 형태이고 암호화되어 있지 않다.

4) 요청 재구성을 위한 파라미터 획득

잔디는 데이터를 조회 및 다운로드하는 API에 대해 HTTP 요청을 재구성하여 Table 8.과 같이 데이터를 획득할 수 있다.

Table 8. JANDI's parameter acquisition path

id	Data acquired	Acquisition Path
1	Token	volatile memory
2	teamID	API 1
3	roomID	API 2
4	fileID	API 4

API 요청에서 잔디는 사용자를 인증하기 위해 하나의 토큰을 사용하며, 이 토큰은 잔디 애플리케이션이 구동되고 있는 환경에서 메모리상에 존재하므로 메모리 덤프 도구를 사용하여 획득해야 한다. 메모리를 덤프하기 위해 오픈소스 메모리 획득 도구인 winPmem을 사용하였다. 해당 도구 사용 시 메모리 덤프 파일을 획득할 수 있는데 해당 파일을 분석하기 위해 HxD가 필요하다.

HxD로 덤프 된 메모리 데이터를 확인 시 Fig. 4.와 같이 '_jd_.access_token=' 형태의 문자열을 확인할 수 있으며, 해당 문자열부터 세미콜론 앞까지 이어지는 문자열이 인증 토큰이다.

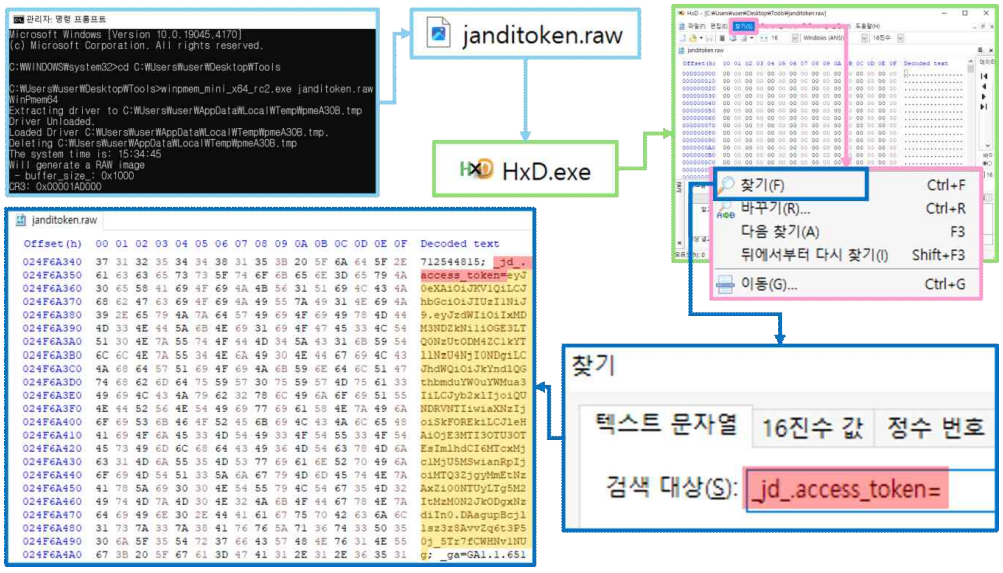

Fig. 4. Authentication token acquisition procedure

채팅 데이터와 송수신한 파일을 얻기 위해 Fig. 5.와 같이 먼저 획득한 인증 토큰과 API-1을 통해 사용자가 속해 있는 teamID를 획득해야 한다.

Fig. 5. Acquire teamID through API-1

이렇게 획득한 teamID와 인증 토큰을 사용하여 API-2를 통해 Fig. 6. 과 같이 사용자가 참여 중인 모든 채팅방의 roomID를 획득한다.

Fig. 6. Acquire roomID through API-2

이후 선행 과정에서 획득한 인증 토큰, teamID 및 roomID를 API-3에 대한 요청 시 활용하면 Fig. 7.과 같이 특정 채팅방에 대한 모든 채팅 데이터를 획득할 수 있다.

Fig. 7. Acquisition of conversation content through API-3

또한 인증 토큰과 teamID를 기반으로 Fig. 8.과 같이 API-4를 통해 사용자가 참여 중인 모든 대화방에서 공유된 파일 목록을 확인할 수 있다. 해당 과정에서 파일 리스트 데이터 내에 특정 파일의 fileID 또한 식별할 수 있다.

Fig. 8. Acquire file list through API-4

마지막으로 API-4에서 획득한 fileID를 사용하면 Fig. 9.와 같이 API-5를 통해 특정 파일에 대한 다운로드 URL을 획득할 수 있다.

Fig. 9. Acquisition of file download URL through API-5

5) 가상 시나리오

국내 화장품 기업 A사의 연구원 김 실험은 신제품 스킨케어 개발 프로젝트의 핵심 인물로, 신제품에 들어갈 새로운 유효성분을 개발하고 있었다. A사는 보안이 철저하게 관리되는 회사였으며, 직원들은 출근 시 개인 스마트폰을 회사에 맡기고, 업무용으로는 회사에서 제공한 노트북만을 사용했다. 이 노트북에는 메신저 애플리케이션인 '잔디'만 설치되어 있었고, 사내에서의 소통은 오직 잔디를 통해서만 이루어졌다. 그런데 2주 후, A사의 경쟁사인 B사에서 동일한 유효성분을 사용한 제품이 출시되었고, A사 내부에서는 기술 유출에 대한 의혹이 불거졌다. 특히, 김 실험과 친분이 있던 두 명의 동료(김 과제, 김 연구)가 최근 갑작스럽게 B사로 이직한 것이 의심스러웠다. 더욱이, 김 실험은 퇴사 직전 고급 승용차와 사치품을 구매하며 소비 패턴이 급격히 변했는데, 이 또한 의혹을 키웠다. 하지만 김 실험의 노트북은 제출 당시 완벽하게 포맷된 상태였다. 그럼에도 불구하고 A사는 경쟁사로 이직한 김 연구의 노트북이 온전한 상태로 남아 있다는 것을 확인하고, 기술 유출의 단서를 찾기 위해 디지털 포렌식 조사를 의뢰한다.

6) 데이터 수집을 통한 대화 내용 재구성

단계별로 API 요청을 재구성하여 필요한 파라미터를 획득한 후 대화 내

용을 재구성해 보면 Fig. 10.과 같다.

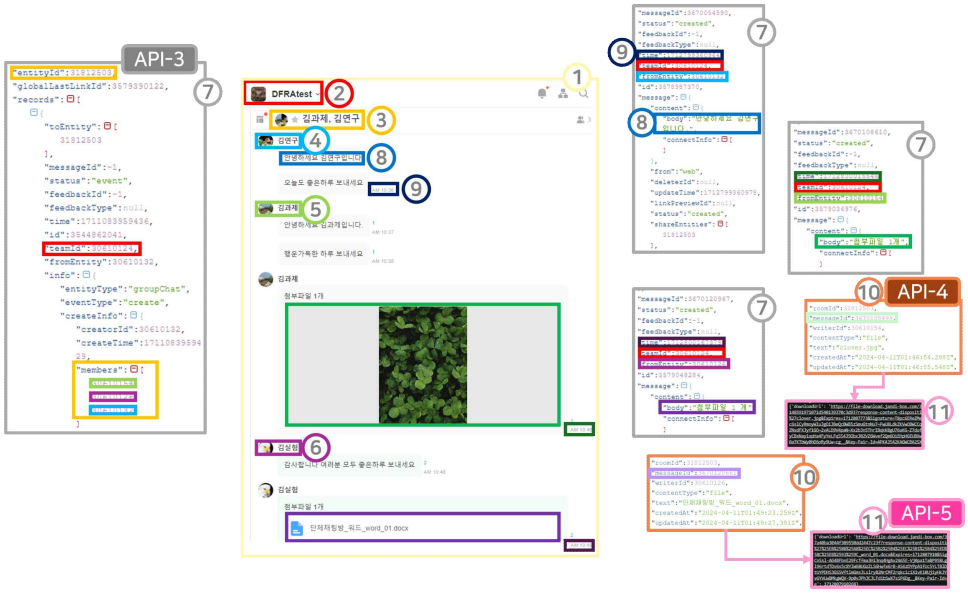

Fig. 10. Reorganize conversation content in group chat room

그림 내 중앙에 연한 노란색 박스 즉 그림의 1번으로 표시된 사진의 모습은, 붉은색 박스 즉 그림의 2번으로 표시된 DFRAtest팀의 단체 대화방이다. 이 대화방은 주황색 박스 즉 그림의 3번으로 표시되어 있으며, 해당 대화방에서는 파란색, 초록색, 자주색으로 표시된 4번 5번 6번인 총 3명의 멤버가 대화하는 상황이다. 대화 요청 재구성을 위해 메모리로부터 얻은 토큰값과 API-1와 API-2를 재구성하여 얻은 teamID와 roomID를 이용하여 회색 박스 즉 7번으로 표시된 API-3로부터 대화 내용을 획득하였다. 이때 메모리에서 인증 토큰을 획득한 후 API 요청 재구성을 진행하기 때문에 타 시스템에서 데이터를 획득 진행이 가능하다. 대화방의 인물 중 하나인 파란색 박스로 표시된 4번 '김연구'와 관련된 대화의 내용은 진한 파란색 박스 즉 그림의 8번으로 표시하였으며 해당 대화가 오간 시간은 남색 박스 즉 그림의 9번으로 표시하였다. 이때 '김연구'와 관련된 식별 정보인 _jd_.member_id는 API 요청 재구성에 필요한 파라미터에는 포함

되지 않지만, 앞서 3장의 2항에서 설명한 것처럼 로컬 아티팩트 분석을 통해 획득 가능한 값으로, 팀 내 각 사용자를 구분하는 고유 식별자이다. 이 값은 API 응답에 포함되는 fromEntity, writerId 등의 값과 직접 대응된다. 따라서 로컬에서 확보한 _jd_.member_id와 API 응답의 fromEntity 값이 일치한다는 것은, 해당 컴퓨터의 사용자가 실제로 이 대화방에 참여하고 있었음을 확인할 수 있는 근거가 된다. 이 대화방은 단체대화방이므로 '김 연구' 이외에도 초록색과 자주색 박스로 표시된 5번 6번의 '김 과제'와 '김 실험'은 각각 첨부파일로 사진과 워드 파일을 보낸 것이 확인되어 갈색 박스로 표시된 10번의 API-4에서 얻은 fileID를 API-5에 넣어 최종적으로 분홍색 박스 즉 11번으로 표시된 파일을 다운로드할 수 있는 URL을 획득하였다. 이러한 과정을 통해 잔디의 전체 대화 내용 재구성 및 송수신된 파일을 획득할 수 있다.

제 5 장 비 설치형 클라우드 서비스

제 1 절 네이버톡

1) 크리덴셜 획득

네이버톡이 웹을 통해 실행 중인 경우, 웹 브라우저의 개발자 도구를 연후, Application 패널에서 Storage 섹션의 Cookies에 접근한 뒤, Cookies 하위에 위치한 'https://ntalk.naver. com' 도메인의 항목 중 key 값이 'NID_AUT', 'NID_SES' 로 표시된 항목의 Value 값에 Fig. 11.과 같이 크리덴셜 데이터가 존재한다.

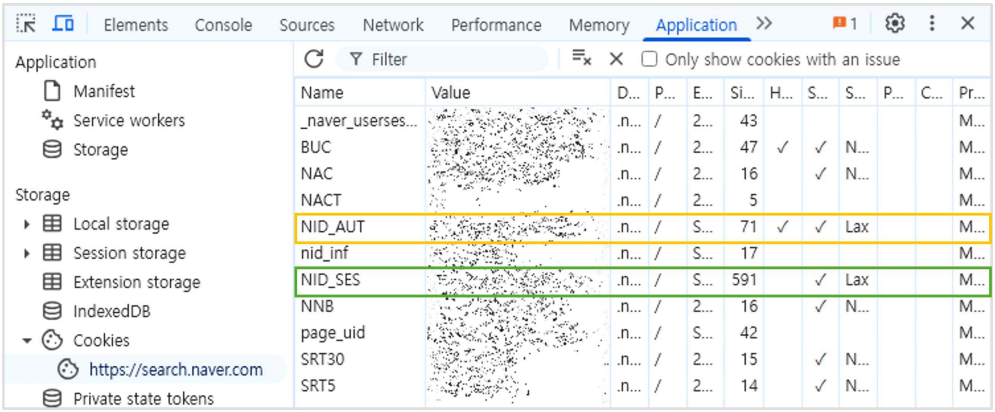

Fig. 11. Obtaining NaverTalk authentication tokens through developer tool

웹 브라우저에서 확인된 'NID_AUT', 'NID_SES'값은 해당 서비스가 실행 중인 환경의 메모리 덤프에서도 Fig. 12.와 같이 동일하게 잔존한다.

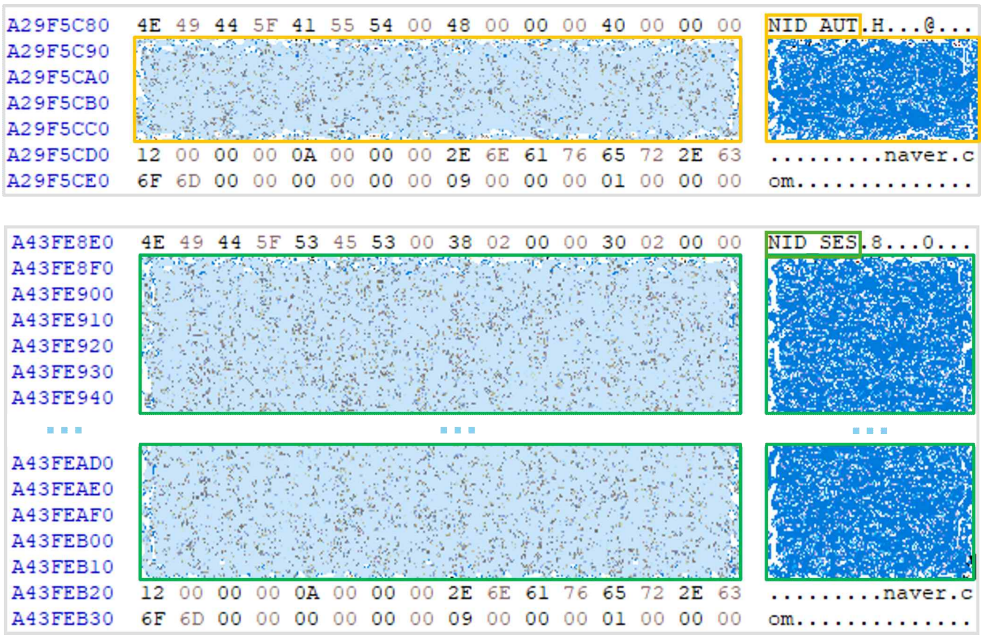

Fig. 12. Obtaining NaverTalk authentication tokens through memory dump

2) 클라우드 데이터 수집

네이버톡은 크게 공개톡, neotalk, 그리고 talktalk으로 구성되어 있다. 공개톡은 동일한 관심사를 가진 다수가 참여할 수 있는 채팅 서비스로, 로그인 없이 이용할 수 있으며 일반 메신저의 오픈채팅 기능과 유사하다. 반면 neotalk과 talktalk은 로그인이 필요한 메신저 서비스이다. 두 서비스는 모두 메신저 기능을 제공하지만 활용 목적과 통신방식에서 차이가 있다. 서비스의 활용 목적을 기준으로 보면, neotalk은 오픈채팅·그룹톡·1:1 DM 등 일반 메신저와 유사한 기능을 제공하는 네이버톡의 일반 대화방이며, talktalk은 상점·판매자와의 상담 및 주문·배송 관련 대화에 특화된 서비스이다. 한편 통신 방식의 측면에서는, talktalk이 HTTP 기반 폴링(Polling)을 사용하여 서버에 일정 주기로 계속 요청을 보내 통신하는 반면, neotalk은 웹소켓(WebSocket)을 사용하여 변경 사항을 실시간으로 처리한다. 본 논문에서는 일반 메신저와 유사한 기능을 제공하는 neotalk에 초점을 맞추어, 그 채팅 데이터를 획득하기 위한 방법을 기술한다. neotalk

의 채팅 데이터를 수집하려면 먼저 NID_AUT와 NID_SES라는 두 개의 쿠키 값, 웹소켓 통신에 사용되는 사용자 식별자인 UID(memberKey), 채팅방의 고유 식별자인 cid(channelId), 그리고 인증 토큰인 accTkn (accessToken)을 확보해야 한다. UID와 accTkn은 “https://apis.naver.com/noc-web/noc-view-api/v1/env”에 HTTP GET 요청 시 Header에 NID_SES와 NID_AUT 값을 포함해 보내면, 응답 JSON의 memberKey 및 accessToken 항목에서 확인할 수 있다. cid 또한 “https://talks.naver.com/api/mytalk/my”에 HTTP GET 요청 시 Header에 NID_SES와 NID_AUT 값을 포함해 보내면, 응답 JSON의 channelId에서 확인할 수 있다. 이후 채팅 데이터를 획득하기 위한 웹 소켓 통신이 수행한다. 각 단계는 Table 9.와 같이 진행된다.

Table 9. Websocket communication for obtaining chat data

Request	Response

{
“ver”: “3”,
“cmd”: 100,
“svcid”: “openchat”,
“cid”: [CID],
“bdy”: {
“uid”: [UID]
“accTkn”: [accTkn] }
	{
	…
	“retMsg”:“SUCCESS
	“sid”:[SID]
	…
	}

{
“ver”: “3”,
“cmd”: 5008,
“svcid”: “openchat”,
“cid”: [CID],
“sid”: [SID],
}
	{
	…
	“retMsg”:“SUCCESS
	…” }

{
“ver”: “3”,
“cmd”: 5003,
“svcid”: “openchat”,
“cid”: [CID],
“sid”: “[SID]”,
“bdy”:
{ “targetMessageNo”: None,
“recentMessageCount”: [Message count]
},
}
	[Chat data response value in JSON format]

웹소켓 서버 주소는 기본적으로 “wss://kr-ss1.chat.naver.com/chat”이

며, 이 외에도 kr-ss2,kr-ss3,kr-ss4또한 사용할 수 있다. 웹소켓 통신 과정은 크게 초기 연결 절차와 그 이후의 데이터 요청·응답 과정으로 나눌 수 있다. 초기 연결 절차에서는 서버와 클라이언트 간에 필요한 식별자와 인증 정보가 교환되며, 이어지는 요청에서는 대화 내용이나 감정 데이터 등 구체적인 채팅 데이터를 획득하게 된다. 이때 중요한 점은 첫 번째 요청에 대한 응답에 두 번째 요청에 활용되는 sid 값이 포함된다는 것이다. 이러한 절차가 정상적으로 진행되면 최종적으로 cid값에 해당하는 채팅방의 대화 내용을 JSON 형식으로 응답받을 수 있다. 또한, 가져올 메시지의 개수를 지정하기 위해 recentMessageCount라는 값을 함께 전송할 수 있다. recentMessageCount에 임의의 양의 정숫값을 전송 시 가장 최근 메시지부터 역순으로 N+1개가 반환된다. 따라서 값이 0을 입력하면 가장 최신 메시지 1개만 반환되고, 값이 3일 때는 최신 메시지부터 역순으로 4개의 메시지가 반환된다. 응답으로 온 JSON 데이터는 Table 10.과 같다.

Table 10. Contents of the messageList sub-kit value

Key	Value
channelId	Chat room unique ID
userId	User ID
profile	Nickname and profile picture url
messageNo	Message order
content	Chat contents
memberCount	Number of members in the chat room
messageTypeCode	Message type
messageStatusType	Message status
extras	Additional information
emotion	showing emotion for the comment
createTime	Message sent time
updateTime	Message modification time
readCount	Number of people who read the message

ChannelId 키값은 채팅방의 고유 ID를 나타내며, 이는 API 요청 시 사용되는 cid 값과 동일한 값을 가진다. UserId 키값은 사용자의 고유한 ID를 의미하며, API 요청 시 사용되는 uid 값과 동일한 값을 가진다. 또한

UserId는 선택한 채팅방의 유형이나 프로필 설정에 영향받지 않는다. 기본 프로필이나 익명 프로필을 선택해도 UserId는 항상 동일하기 때문에, 일관된 사용자 추적이 가능하다. profile 키값은 1대1 채팅방에서 name과 profile이라는 내부 키값을 가지며, 각 키에 프로필 명과 사진 URL이 저장되지만. 그룹 채팅에서는 profile 키값이 null 값으로 존재한다. MessageNo 키값은 채팅방 내 메시지의 순서를 나타내며, 방이 개설되었음을 알리는 시스템 메시지가 1로 설정되고 이후 순차적으로 증가한다. 그리고 Content 키값에는 메시지의 내용이 저장된다. MemberCount 키값은 채팅방의 멤버 수를 나타내며, 멤버가 추가될 때마다 증가하게 된다. MessageTypeCode 키값은 메시지의 유형을 나타내며, 101은 채팅방 개설이나 새로운 멤버의 입장을 의미하고, 110은 권한 부여를 나타내며, 1은 일반 메시지, 11은 사진 첨부 메시지, 19는 투표 기능 메시지를 의미한다. MessageStatusType 키값은 메시지의 상태를 나타내는데, “NORMAL”은 시스템 알림, 일반 메시지, 투표 내용 등을 포함한 모든 종류의 메시지에 기본적으로 부여되는 상태이다. 반면, “RECLAIM”은 사용자가 5분 이내에 본인의 메시지를 삭제한 경우에 부여되고, “HIDDEN”은 방장이나 부방장이 5분 이후에 본인을 포함한 다른 사람의 메시지를 삭제한 경우에 나타난다. Extras 키값은 순수 문자열로만 이루어진 메시지 전송에서는 값이 존재하지 않지만, 사진 첨부 메시지, 시스템 알림, 투표 등의 다양한 메시지에서는 해당 특징에 따른 정보를 포함한다. Extras 키값은 기본적으로 순수 문자열로만 이루어진 일반 메시지에서는 존재하지 않는다. 그러나 예외적으로 순수 문자열 메시지라 하더라도 다른 메시지에 대한 답장일 경우에는 extras에 replyOriginMessage가 포함되어, 답장 대상 메시지의 번호, 내용, 발신자 정보 등이 기록된다. 이 외에도 사진이나 파일 첨부, 시스템 알림, 투표와 같이 부가적인 기능이 수반되는 메시지의 경우, extras는 해당 기능을 설명하는 구조화된 데이터를 포함한다. Emotion 키값은 메시지에 공감이 표시된 경우에만 존재한다. 메시지에 공감이 표시된 경우 emotionTypeCodeList는 공감의 종류를 나타내며, count는 공감의 개수를 의미한다. 이 기능은 카카오톡에서 하트나 엄지손

가락으로 공감을 표시하는 방식과 유사하다. CreateTime 키값은 메시지가 전송된 시간을 나타내고, UpdateTime 키값은 메시지가 수정된 시간을 나타낸다. 초기에는 createTime과 동일한 값을 가지지만, 삭제가 발생하면 삭제된 시간으로 변경된다. 마지막으로, ReadCount 키값은 메시지를 읽은 사람의 수를 나타내며, 기본값은 1로 설정된다. 이후 누군가 메시지를 읽으면 1씩 증가하며, '이 사람 메시지 숨기기' 기능을 이용해도 읽음 수는 증가하게 된다.

3) 안티 포렌식 행위에 따른 클라우드 데이터 변화 분석

네이버톡에는 '이 사람 메시지 숨기기'와 '삭제하기'라는 두 가지의 안티 포렌식 기능이 존재한다. 먼저, 특정 메시지에 대해 '이 사람 메시지 숨기기' 기능을 사용할 경우 해당 메시지뿐만 아니라, 그 사용자가 지금까지 전송한 모든 메시지가 채팅방 화면에서 사라진다. 이 효과는 24시간 동안만 유지되며 해당 채팅방에 한정되고, 취소할 수 없다. 그러나 이러한 숨김은 단순히 사용자 인터페이스(UI) 상에서만 적용되는 것이므로, HTTP GET 요청을 통해 대화 내용을 획득할 경우 해당 메시지의 content 키값은 여전히 존재한다. 반면, '삭제하기' 기능을 사용할 경우 선택한 메시지의 내용은 "삭제된 메시지입니다."라는 문구로 대체된다. 삭제된 메시지는 숨김 처리 기능과는 달리, 별도로 숨길 수 없다. 또한 해당 메시지는 서버에서도 삭제되므로, HTTP GET 요청을 통한 대화 내용 획득 시 extras와 content 키값은 비어 있는 값으로 표시된다. 이때 메시지의 수정 시간을 나타내는 UpdateTime 값은 삭제 시각으로 변경된다. 추가적으로 MessageStatusType 값이 "RECLAIM"으로 설정된 경우는 사용자가 5분 이내에 자신의 메시지를 삭제한 경우이며, "HIDDEN"으로 설정된 경우는 방장이나 부방장이 특정 사용자의 메시지를 삭제한 경우에 해당한다. 따라서 content 값이 존재하지 않고, 동시에 MessageStatusType이 특정 상태로 기록되며, UpdateTime 값이 CreateTime 값과 다른 경우 해당 메시지가 삭제된 것으로 판단할 수 있다.채팅방 유형 및 권한 그리고 안티 포

렌식 기능별로 분류한 내용은 Table 11.과 같다.

Table 11. Summary of Naver Talk's Anti-Forensic Behavior

<table>
<tr><th colspan="2" rowspan="2"></th><th colspan="2">1:1</th><th colspan="2">group</th></tr>
<tr><th>Self</th><th>Others</th><th>Self</th><th>Others</th></tr>
<tr><td rowspan="2">Authority O</td><td>Delete</td><td colspan="2" rowspan="2">'Authorization' does not exist</td><td>O</td><td>O</td></tr>
<tr><td>Hide</td><td colspan="2">concurrently</td></tr>
<tr><td rowspan="2">Authority X</td><td>Delete</td><td>Within 5 minutes</td><td>X</td><td>Within 5 minutes</td><td>X</td></tr>
<tr><td>Hide</td><td>X</td><td>X</td><td>X</td><td>O</td></tr>
</table>

먼저, 1:1 채팅의 경우 방장이나 부방장과 같은 권한 구조가 존재하지 않는다. 이 환경에서는 사용자가 본인이 작성한 메시지만 삭제 할 수 있으며, 삭제는 전송 후 5분 이내에만 가능하다. 해당 메시지가 삭제되면 데이터베이스 상에서는 extras와 content 값이 공란으로 표시되고, UpdateTime은 삭제된 시각으로 변경된다. 사용자 인터페이스(UI)에서는 말풍선 내 메시지가 "삭제된 메시지입니다."라는 문구로 대체되며, 이때 MessageStatusType 값은 "RECLAIM"으로 설정된다. 반면, 숨기기 기능은 제공되지 않는다. 단체 채팅에서 권한이 없는 사용자는 두 가지 상황으로 나눌 수 있다. 첫째, 상대방의 메시지에 대해 '이 사람 메시지 숨기기' 기능을 적용할 수 있다. 이 경우 해당 사용자가 작성한 모든 메시지는 24시간 동안 현재 채팅방에서만 숨겨지며, 이는 UI 상에서만 반영될 뿐 데이터베이스의 content 값은 그대로 존재한다. 둘째, 본인이 작성한 메시지를 삭제 할 경우에는 1:1 채팅과 동일하게 전송 후 5분 이내에만 가능하다. 이때 역시 extras와 content 값은 공란으로 처리되고, UpdateTime은 삭제된 시각으로 변경되며, UI에서는 "삭제된 메시지입니다."로 표시된다. MessageStatusType은 "RECLAIM"으로 설정된다. 숨기기 기능은 지원되지 않는다. 반면, 단체 채팅에서 방장과 부방장 같이 권한이 있는 사용자는 보다 강력한 제어 권한을 가진다. 상대방의 메시지를 삭제 할 수 있으

며, 이 경우 데이터베이스 상에서는 extras와 content 값이 공란으로 처리되고, UpdateTime은 삭제된 시각으로 변경된다. UI 상에서는 삭제와 동시에 숨김 처리가 이루어지며, MessageStatusType 값은 “HIDDEN”으로 변경된다. 따라서 삭제와 숨김 기능을 따로 적용할 수 없다. 또한 권한이 있는 사용자는 특정 상대방의 메시지에 대해 ‘이 사람 메시지 숨기기’ 기능을 사용할 수 있으며, 이 경우 해당 사용자의 모든 메시지가 24시간 동안 해당 채팅방에서만 숨겨진다. 한편, 권한이 있는 사용자가 본인의 메시지를 삭제 할 경우에도 동일하게 extras와 content 값이 공란 처리되고, UpdateTime은 삭제된 시각으로 변경된다. UI에서는 삭제와 동시에 숨김 처리가 이루어지며, MessageStatusType 값은 “HIDDEN”으로 기록된다. 그러나 이 경우에는 별도의 숨기기 기능을 적용할 수 없다.

4) 클라우드 포렌식 절차 및 검증

네이버톡의 크리덴셜 확보는 우선 웹 환경에서 실행 중인 네이버톡의 주요 쿠키값(NID_AUT, NID_SES)을 획득하는 것으로 시작한다. 이 값은 웹 환경에서 실행 중인 네이버톡을 대상으로 웹 브라우저의 개발자 도구를 통해 직접 추출하거나, 서비스가 동작 중인 환경의 메모리 덤프를 분석하여 확보할 수 있다. 다음 단계로, 채팅 데이터 수집에 필요한 사용자 식별자인 UID, 채팅방의 고유 식별자인 cid, 인증 토큰인 accTkn 를 확보한다. 이를 위해 Table 12.에 해당하는 API에 앞서 획득한 쿠키값을 HTTP GET 요청의 헤더에 포함하여 전송하고, 응답으로 반환되는 JSON 데이터에서 해당 값을 추출한다.

Table 12. URL by ‘Value to be Obtained’

Value	URL
UID(memberKey)	https://apis.naver.com/noc-web/noc-view-api/v1/env
accTkn(accessToken)	
cid(channelId)	https://talks.naver.com/api/mytalk/my

마지막으로 확보한 UID, cid, accTkn을 이용해 웹소켓 서버와 연결을 구성한다. 연결에 사용되는 서버 주소는 wss://kr-ss1.chat.naver.com/chat이다. 이를 통해 실시간으로 전송되는 채팅 데이터를 JSON 형식으로 수집할 수 있다. 수집된 데이터는 단순 로그 분석에 그치지 않고, Fig. 13.과 같이, 채팅방 형태로 재구성할 수 있다.

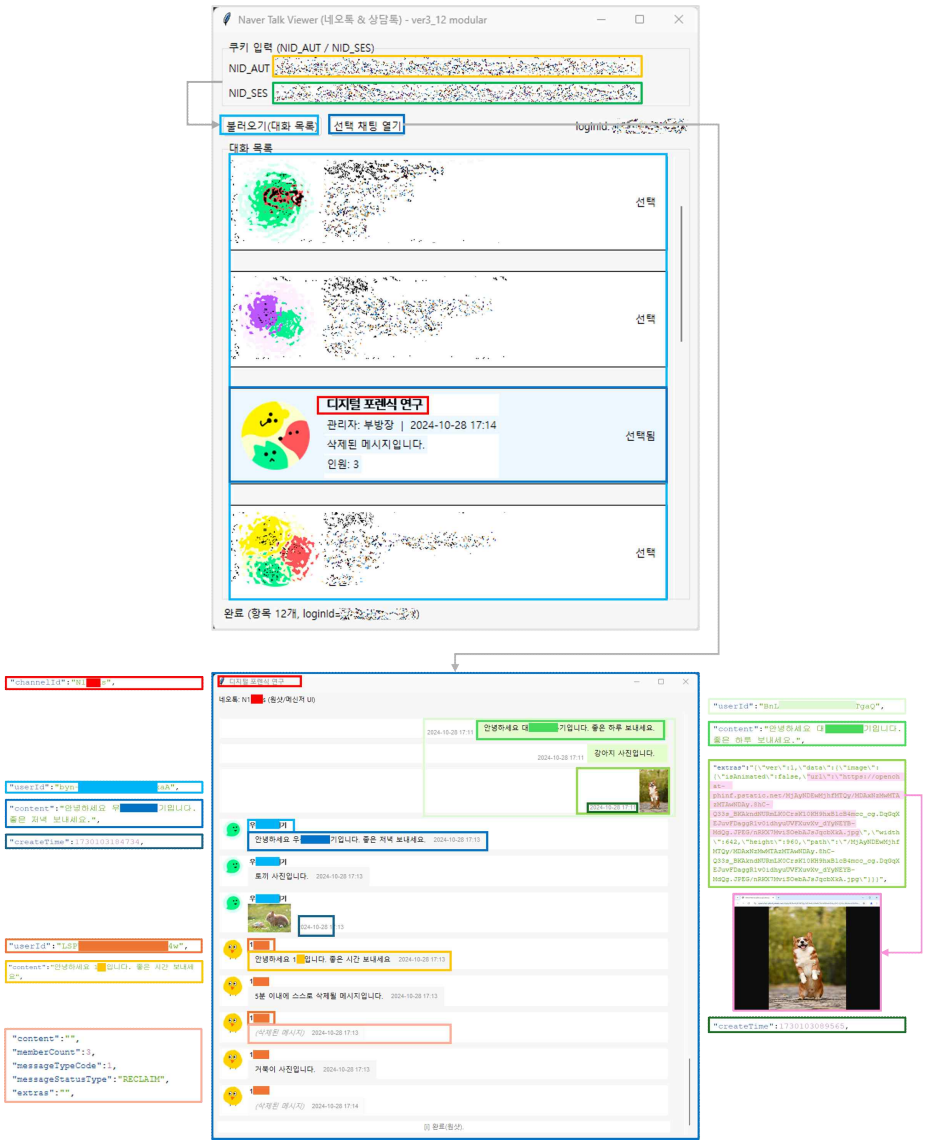

Fig. 13. Reconstruction of Naver Talk chat room contents

본 연구에서는 이를 검증하기 위해 전용 도구를 활용하여 채팅방 화면을 재현하였다. 이러한 과정은 제안한 데이터 수집 방법이 실제 환경에서 실질적으로 동작할 수 있음을 보여준다.

제 2 절 디스코드

1) 크리덴셜 획득

디스코드가 웹을 통해 실행 중인 경우, 웹 브라우저의 개발자 도구를 연후, Application 패널에서 Storage 섹션의 Local storage에 접근한 뒤, Local storage 하위에 위치한 'https://discord.com' 도메인의 항목 중 key 값이 'token'으로 표시된 항목의 Value 값에 Fig. 14.와 같이 크리덴셜 데이터가 존재한다.

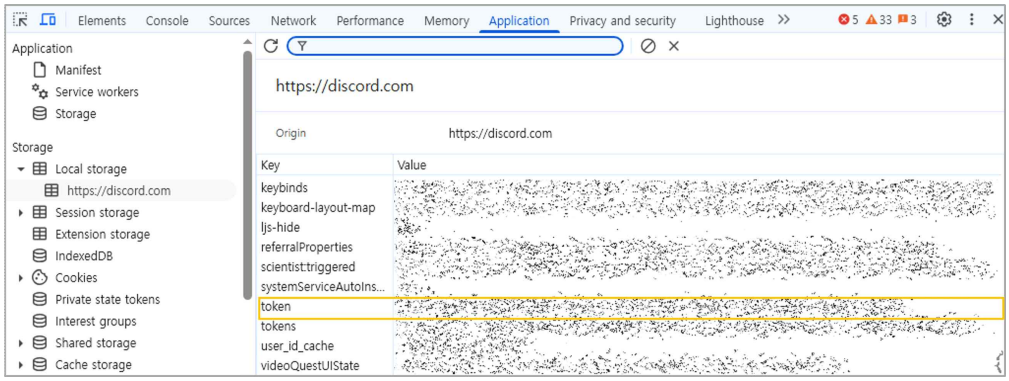

Fig. 14. Obtaining Discord tokens through developer tool

웹 브라우저에서 확인된 'token' 값은 해당 서비스가 실행 중인 환경의 메모리 덤프에서도 'authorization'이라는 값으로 Fig. 15.와 같이 동일하게 잔존한다.

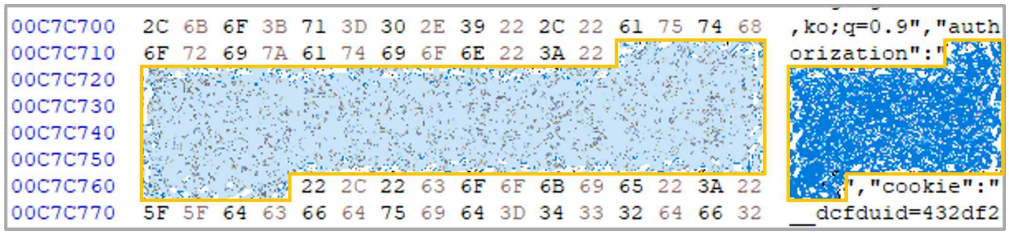

Fig. 15. Obtaining Discord tokens through memory dump

2) 클라우드 데이터 수집

디스코드는 크게 서버와 다이렉트 메시지로 구분된다. 서버는 다수의 사용자가 공통의 주제나 목적을 중심으로 모이는 공간으로, 그 안에는 채팅 채널과 음성 채널이 존재한다. 채팅 채널은 일반적인 메신저의 오픈채팅 기능과 유사하며, 음성 채널은 실시간 음성 통화 및 영상 통화 기능을 제공한다. 다음으로 다이렉트 메시지는 일반 메신저의 1:1 채팅과 유사한 기능을 가지며, 텍스트 기반 메시지 전송뿐만 아니라 음성 및 영상 통화도 지원한다. 정리하면, 디스코드는 구조적으로 서버와 다이렉트 메시지로 구분되고, 기능적으로는 텍스트 기반 채팅과 음성 기반 대화로 나눌 수 있다. 이중 텍스트 기반 채팅은 사용자의 활동과 행위를 직접적으로 반영하는 중요한 포렌식 증거가 되므로, 본 논문은 서버 내 채팅 채널과 다이렉트 메시지에서 생성되는 채팅 데이터를 대상으로 데이터 획득 방법을 기술한다. 디스코드의 채팅 데이터를 수집하려면 먼저 token값을 확보해야 한다. 이후 token값을 포함하여 관련 HTTP API를 호출하면 사용자 정보, 특정 유저 정보, 사용자의 친구 목록, 특정 유저 프로필 사진, 다이렉트 메시지 채널 목록, 사용자가 속한 서버 목록, 서버 내 채널 목록, 채널 내 메시지 조회 등을 응답으로 반환한다. 수집에 활용한 HTTP API 요청 URL은 Table 13.과 같다.

Table 13. Classifying request and response data for data acquisition

URL	Response
/users/@me	User Information
/users/{user_id}/profile	Specific user information
/users/@me/relationships	User's friends list
/users/@me/relationships/{user_id}/{avatar}.webp?size=160	Specific user profile picture
/users/@me/channels	Direct Message Channel List
/users/@me/guilds	List of servers to which the user belongs
/guilds/{guild_id}/channels	List of channels within the server
/channels/{channel_id}/messages	View messages within a channel

디스코드의 사용자 정보, 특정 유저 정보 그리고 사용자의 친구 목록에 해당하는 URL에 필요한 파라미터를 포함하여 HTTP API 요청을 전달한 결과, 응답으로 온 데이터는 Table 14.와 같다.

Table 14. Classification of response data related to Discord user information, specific user information, and the user's friends list.

Key	Value
id	User Unique ID
username	Unique username
avatar	User Avatar ID
global_name	Displayed user name
email	User Email
bio	User introduction
nickname	A nickname given to that user by the user

응답 데이터에는 공통적으로 id, username, avatar, global_name과 같은 키가 포함된다. 이때 응답 유형별로 추가로 포함되는 키는 다음과 같다. 사용자 정보에는 bio와 email이, 특정 유저 정보에는 bio가, 친구 목록 정보에는 nickname이 각각 추가로 포함된다. 다만, 응답 구조에 따라 이러한 키의 위치가 달라진다. 개인 사용자 정보 조회 결과에서는 공통적으로 존재하는 키값들이 최상위에 위치하는 반면, 특정 사용자 정보와 친구 목록 응답에서는 동일한 값들이 모두 user라는 하위 키 아래에 포함된다. 이러한 구조적 차이가 존재하더라도 실제로 지칭하는 정보의 의미는 동일하다. 각 키의 역할을 살펴보면, 먼저 id는 해당 유저의 고유 식별 값을 의미하며, username값은 고

유 유저명으로 주로 친구 요청을 위한 고유 식별자로 사용된다. avatar는 프로필 사진에 대한 고유 식별 값이고, global_name은 디스코드 내에서 다른 사용자가 인식하는 이름이다. 즉, username은 유저를 구별할 수 있는 고유한 값으로 중복될 수 없지만, global_name은 DM이나 서버에서 메시지를 보낼 때 확인할 수 있는 이름이다. 따라서 global_name은 다른 사용자가 별명을 부여하지 않았다는 전제하에 DM, 서버, 친구 요청에서 보이는 이름이 된다. 또한 bio는 유저가 작성하는 자기소개말로, 카카오톡의 상태 메시지와 유사한 역할을 한다. 마지막으로 nickname은 사용자가 특정 유저에게 직접 부여한 별명으로, 해당 값이 존재하는 경우 DM이나 서버에서 표시되는 이름은 global_name이 아니라 nickname이 우선적으로 반영된다. 추가적으로, id와 avatar는 데이터 식별 용도뿐만 아니라 프로필 사진 요청 URL을 보낼 시에 활용된다. 프로필 사진 요청 URL인 '/users/@me/relationships/[user_id]/[avatar].webp?size=160'에서 user_id 자리에 id 값을, avatar 자리에 avatar 값을 대입하여 요청을 전송하면 해당 사용자의 프로필 이미지를 webp 형식으로 획득할 수 있다.디스코드의 다이렉트 메시지 채널 목록, 사용자가 속한 서버 목록 그리고 서버 내 채널 목록에 해당하는 URL에 필요한 파라미터를 포함하여 HTTP API 요청을 전달한 결과, 응답으로 온 데이터는 Table 15.와 같다.

Table 15. Classifying response data related to group and private chat rooms

URL	Key	Value
Direct Message Channel List	id field inside recipients	User Unique ID
	username field inside recipients	Unique username
	avatar field inside recipients	User Avatar ID
	global_name field inside recipients	Displayed user name
List of servers to which the user belongs	id	Server Unique ID
	name	Server name
List of channels within the server	id	Unique ID of the channel within the server
	type	Channel types within the server
	parent_id	Unique ID of the parent channel
	name	Channel name
	guild_id	Unique ID of the server it belongs to

먼저 다이렉트 메시지 채널 목록 응답에는 사용자 정보, 특정 유저 정보, 친구 목록 응답에서 확인된 id, username, avatar, global_name과 동일한 키가 포함된다. 다만 이 값들은 recipients라는 상위 키의 하위에 존재한다는 구조적 차이가 있다. 이미 앞선 응답에서 설명한 것과 동일하게, 여기서도 id는 유저 고유 ID, username은 고유 유저명, avatar는 프로필 사진을 식별하는 값, global_name은 디스코드에서 표시되는 이름을 의미한다. 다음으로 서버 목록 응답에는 id와 name이 포함되며, 각각 서버의 고유 식별 값과 서버의 이름을 나타낸다. 마지막으로 서버 내 채널 목록 응답에는 id, type, guild_id, name, parent_id가 포함된다. 이 중 id는 채널의 고유 식별 값이고, type은 채널의 유형을 구분하는 값으로 4, 2, 0, 1의 네 가지가 존재한다. type=4는 서버 내 채널 중 채널 분류용 인덱스 채널로 채팅방 역할이 아닌 채널들을 분류하는 역할을 하는 채널이며, type=2는 서버 내 채널 중 음성 채널, type=0은 서버 내 채널 중 메시지를 주고받는 채팅 채널, type=1은 다이렉트 메시지 채널을 의미한다. parent_id는 채널 간의 상하위 관계를 드러내는 값으로, 상위 채널의 고유 식별 값을 의미하며 앞서 언급된 type과 연결되는 값이다. type=2 또는 0인 채널은 각각 음성 또는 채팅방의 역할을 하고 있기 때문에 이 채널들은 type 4를 가진 채널의 하위에 속한다. 따라서 type=2 또는 0인 채널은 parent_id에 본인이 속해있는 상위 채널의 식별 값을 가지고 있으며 type 4를 가진 채널은 parent_id가 null을 가진다. 다음으로 name은 해당 채널의 이름을 의미하며, guild_id는 해당 채널이 속해있는 서버의 식별 값을 의미한다. 서버 내 채널 메시지와 DM 채널 메시지에 해당하는 URL에 필요한 파라미터를 포함하여 HTTP API 요청을 전달한 결과는 Table 16.과 같으며, 두 응답은 기본적으로 동일한 JSON 구조를 가진다.

Table 16. Classification of response data related to server-side channel message queries and DM channel message queries

Key	Value
type	Message type

id field inside components	Message Unique ID
content	Message body
timestamp	Message sent time
edited_timestamp	Time the message was modified
id field inside author	ID of the user who sent the message
author 하위 username	The unique username of the user who sent the message
global_name field inside author	The displayed user name of the user who sent the message
call_ended_timestamp	Time of call made/ hang up
call_participants	ID of the user who participated in the call
id field inside referenced_message	Sticker Message Unique ID
id field inside sticker_items	Sticker ID
sticker_items 하위 name field inside	Name of the sticker
message_id field inside message_reference	The ID of the reply message or the unique ID of the sticker message
id field inside mentions	ID of the user mentioned with @ or the reply target user
username field inside mentions	Unique username of the user mentioned with @ or the reply target user
global_name field insidementions	Displayed username of the user mentioned with @ or the reply target user
content_type field inside attachments	Extension of attached file
id field inside attachments	ID of the attached file
title field inside attachments	File name of the attached file
filename field inside attachments	File name and extension of attached file with Korean characters removed
url field inside attachments	Download link for attached file
size field inside attachments	Size of attached file
width field inside attachments	Width of the attached file
height field inside attachments	Height of attached file

응답에는 공통적으로 ‘type’, ‘components 하위의 id’, ‘content’, ‘timestamp’, ‘edited_timestamp’ ‘author 하위의 id, username, global_name’, ‘referenced_message 하위 id’,‘sticker_items 하위 id, name’, ‘message_reference 하위 message_id’, ‘mentions 하위의 id, username, global_name’, 그리고 첨부파일 관련 키인 ‘attachment 하위 content_type, id, title, filename, url, size, width, height’가 포함된다. 공통적으로 등장한 키 중 type은 메시지의 유형을 나타내는 값으로, 0은 일반 메시지, 3은 통화 관련 메시지, 7은 사용자가 채팅방에 입장한 경우의 메시지, 19는 다른 메시지에 대한 답장을 의미한다. 이때 type=19의 답장은 단순 @를 붙여 상대방을 멘션하는 것이 아닌 답장할 메시지에 우클릭해 답장 기능을 사용했을 때만 19로 표시되며, 단순 @로 멘션은 일반 메시지로 분류된다. 추가적으로 '손을 흔들어 인사해 보세요!' 기능도 답장 기능에 포함되어 type=19로 표현된다. components 하위의 id는 메시지의 고유 식별 값을 의미하며, content는 메시지 본문을 의미한다. 이때 일반적인 텍스트 메시지뿐 아니라 디스코드 내 기능인 GIF 이미지도 이 content에 포함된다. 또한 @로 상대방을 멘션 했을 경우 UI에는 ‘@멘션된 상대방의 표시되는 유저명’, 즉 global_name이 표시되지만, JSON 상에는 ‘@멘션된 상대방의 고유 식별자’, 즉 id 값이 기록된다. timestamp는 메시지가 전송된 시점을 의미하며, edited_timestamp는 메시지가 수정된 시간을 의미하고, author 하위의 id, username, global_name은 각각 메시지를 보낸 사람의 고유 식별 값, 고유 유저명, 디스코드 상에서 표시되는 이름을 의미한다. referenced_message 하위 id는 스티커 메시지의 고유 식별 값을 의미하며, sticker_items 하위 id와 name은 각각 디스코드 스티커 자체의 의 고유 식별자와 이름을 의미한다. 디스코드의 스티커는 카카오톡의 이모티콘과 유사한 기능을 한다. message_reference 하위 message_id는 답장 대상 메시지의 고유 식별 값 또는 스티커 답장의 고유 식별 값으로, 답장 메시지에만 존재한다. mentions 하위의 id, username, global_name은 각각 멘션된 혹은 답장 대상 유저의 정보인 고유 식별 값, 고유 유저명, 표시되는 유저명을 의미하며, type=19일 때에는 답장 대상의 정보를, 일반 메시

지에서의 멘션일 경우에는 멘션된 대상의 정보를 나타낸다. attachments 하위의 첨부파일 관련 키들은 각각 다음과 같은 의미를 가진다. content_type은 첨부된 파일의 확장자를 나타내며, 이미지, 비디오, Microsoft Office 문서 확장자 등 일부 확장자만 식별이 가능하다. 그러나 한글 전용 확장자인 .hwpx는 확인되지 않았으며, 일부 한글 파일 첨부 시에는 값이 존재하지 않는 경우도 있었다. 이어서 id, title, filename은 첨부파일의 고유 식별 값, 파일명, 한글이 제거된 파일명과 확장자를 의미한다. 이때 실제 파일명을 확인하기 위해서는 두 값을 조합해야 하는데, 이는 title이 한글이 포함된 원본 파일명을 보여주지만 확장자를 포함하지 않는 반면, filename은 확장자를 표시하되 파일명에서 한글이 제거되어 있기 때문이다. 마지막으로 url, size, width, height는 첨부파일의 다운로드 링크, 첨부파일의 크기, 이미지나 동영상을 첨부했을 경우 해당 미디어의 가로 및 세로 크기를 의미한다. 또한 url의 다운로드 링크는 의미하며 일정 시간이 지나면 만료된다. 이와 같이 서버 내 채널 메시지와 DM 채널 메시지는 대부분 동일한 구조와 키를 공유하지만, 세부적으로는 채팅 환경의 차이로 인해 특정 값이나 키에서 차이를 보인다. 구체적으로 이러한 차이는 채팅 환경의 특성에 따라 값의 유형에서 나타나기도 하고, 특정 키의 존재 여부에서 확인되기도 한다. 먼저 값의 유형 측면에서는 type 키에서 차이가 드러난다. 앞서 언급했듯 type은 네 가지 유형을 가지지만, 서버 내 채널에서는 음성 채널이 별도로 존재하기 때문에 DM 채널 메시지에서만 type=3이 나타나고, 반대로 1:1 대화인 DM 채널에서는 type=7이 나타나지 않는다. 다음으로 키의 존재 여부 측면에서는 call 하위 ended_timestamp와 participants가 대표적이다. 이는 각각 통화 종료 시각과 참여자 ID를 의미하는데, DM 채널에는 통화 기능이 존재하기 때문에 포함되지만, 서버 내 채널에는 음성 채널이 존재하므로 해당 키는 나타나지 않는다. 결론적으로 두 환경은 동일한 구조를 공유하면서도, 환경적 차이에 의해 값의 유형과 키의 포함 여부에서 상이한 양상을 보인다.

3) 안티 포렌식 행위에 따른 클라우드 데이터 변화 분석

디스코드에는 '메시지 수정하기'와 '메시지 삭제하기'라는 두 가지의 안티 포렌식 기능이 존재한다. 먼저 '메시지 수정하기'는 사용자가 보낸 메시지의 내용을 수정하는 기능이다. 수정된 파일은 UI상에서는 '수정됨' 이라고 표시되며, content값이 수정된 내용으로 변경되고 edited_timestamp 값이 null에서 수정된 시간으로 변경된다. 이 경우 수정 이전의 메시지 내용은 확인할 수 없지만, 메시지가 언제 수정되었는지는 기록을 통해 파악할 수 있다. 다음으로 ' 메시지 삭제하기'는 보낸 메시지를 삭제하는 기능으로, 이렇게 삭제된 메시지는 더 이상 HTTP API 요청을 통해 획득할 수 없으며, 서버상에서도 삭제가 된다.

4) 클라우드 포렌식 절차 및 검증

디스코드의 데이터 수집은 먼저 token값을 확보하는 것에서 시작된다. 이 값은 웹 환경에서 실행 중인 디스코드를 대상으로 웹 브라우저의 개발자 도구를 통해 직접 추출하거나, 서비스가 동작 중인 환경의 메모리 덤프를 분석하여 확보할 수 있다. 확보된 token값은 이후 사용자 정보, 특정 유저 정보, 사용자의 친구 목록, 특정 유저 프로필 사진, 다이렉트 메시지 채널 목록, 사용자가 속한 서버 목록, 서버 내 채널 목록, 채널 내 메시지 조회 등 다양한 API 요청에 포함되어 데이터 획득의 핵심 매개체로 활용된다. 먼저 사용자 정보의 경우 '사용자 정보' URL에 token 값을 포함하여 요청하면 확인할 수 있으며, 친구 목록 또한 동일하게 '사용자의 친구 목록' URL에 token 값을 넣어 요청하여 획득할 수 있다. 특정 유저 정보는 친구 목록에서 추출한 id 값을 '특정 유저 정보' URL에 token과 함께 추가하여 얻을 수 있으며, 특정 유저 프로필 사진은 해당 유저의 id와 avatar 값을 '프로필 사진' URL에 포함해 요청함으로써 획득된다. DM 채널 목록과 서버 목록 역시 각 URL에 token 값을 포함하여 요청할 수 있으며, 서버 내 채널 목록은 서버 목록에서 얻은 채널 id를 함께 추가해 요청해야 한다. 마지막으로 채널 내 메시지 조회는 DM 채널 목록과 서버 내 채널 목록에서 확인한 채널 id를 이용하여, 해당 id와 token 값을 함께 요청에 포함해 획득할 수 있다. 이러한 일련의 데이터 획득 절차를 도구를 통해 시각적으로 재구성한 결과는 Fig. 16.과 같다.

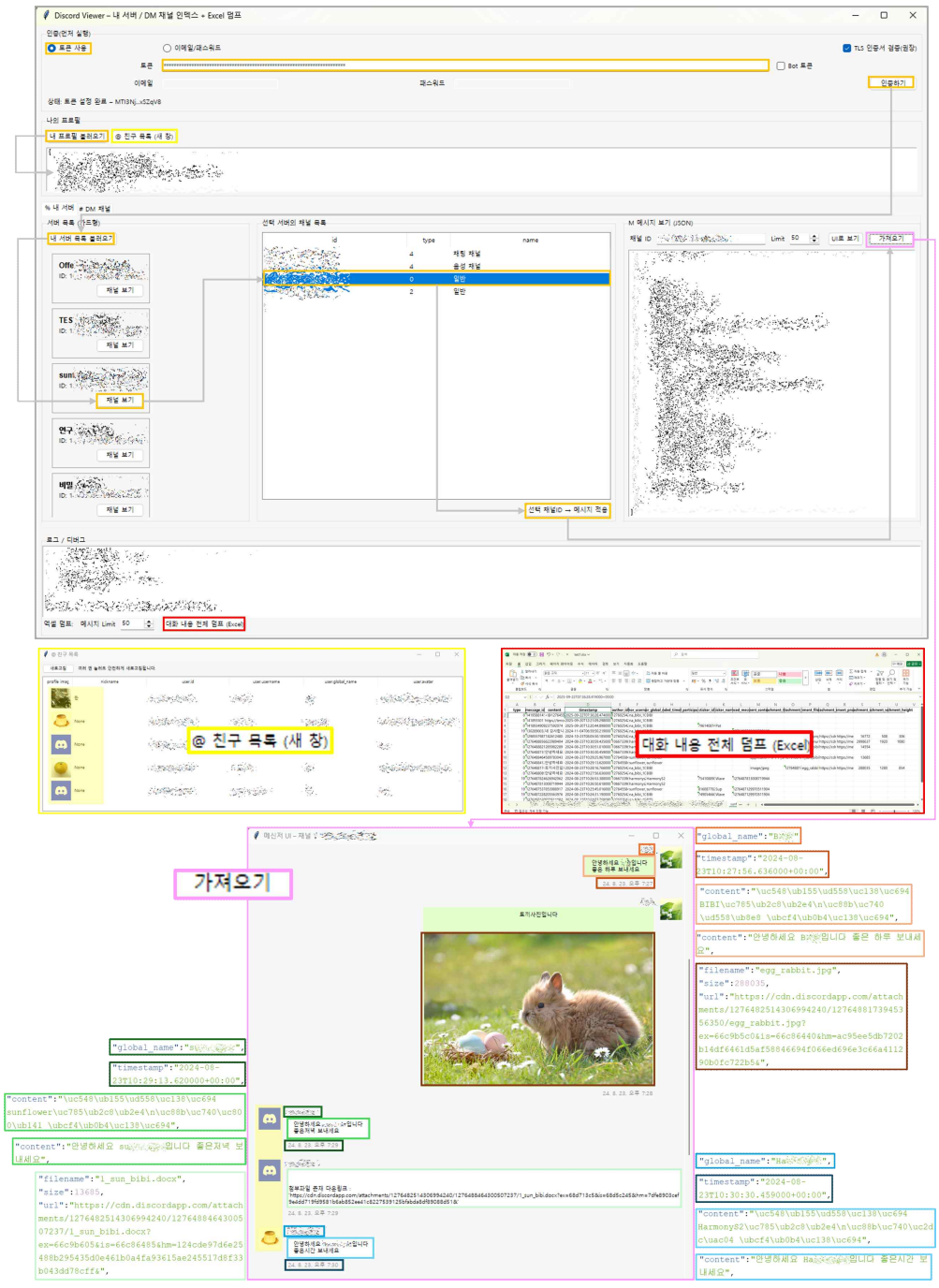

Fig. 16. Reconstructing the chatroom content of a channel within a Discord server

위와 같이 본 연구에서는 도구를 통해 데이터 수집 과정을 실제로 검증하여 사용자 정보, 서버 및 DM 채널의 구조, 그리고 채널 내 대화 내용을

재현할 수 있었다. 이로써 디스코드의 데이터 수집 절차 역시 실효성이 있음을 입증하였다.

제 3 절 폴라리스 오피스

1) 크리덴셜 획득

폴라리스 오피스가 웹을 통해 실행 중인 경우, 웹 브라우저의 개발자 도구를 연 후, Application 패널에서 Storage 섹션의 Cookies에 접근한 뒤, Cookies 하위에 위치한 'https://www.polarisoffice.com' 도메인의 항목 중 key 값이 'SID' 로 표시된 항목의 Value 값에 Fig. 17.과 같이 크리덴셜 데이터가 존재한다.

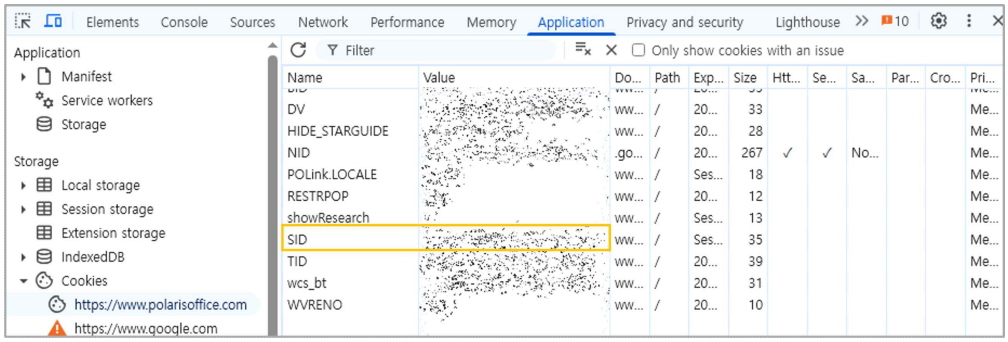

Fig. 17. Obtaining Polaris Office SID through developer tool

웹 브라우저에서 확인된 'SID' 값은 해당 서비스가 실행 중인 환경의 메모리 덤프에서도 Fig. 18.과 같이 동일하게 잔존한다.

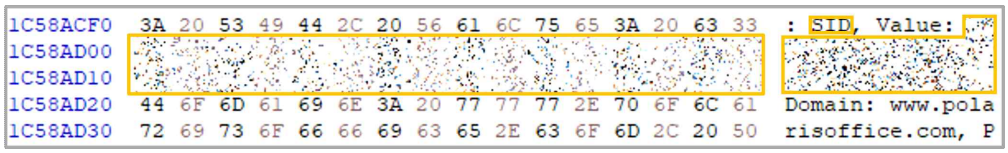

Fig. 18. Obtaining Polaris Office SID through memory dump

2) 클라우드 데이터 수집

본 연구에서는 메신저 서비스 외의 클라우드 기반 응용 프로그램을 분석 대상으로 확장하기 위해 오피스 소프트웨어인 폴라리스 오피스를 선정하였다. 폴라리스 오피스는 클라우드형, PC 평생 소장형, 모바일 클라우드형, iOS 모바일 평생형 등으로 구분되며, 이 중 My Polaris Drive를 사용할 수 있는 것은 클라우드형과 웹 기반 서비스 등 로그인 기반 제품군에 한정된다. My Polaris Drive는 Windows, Mac, Android, iOS, 웹 등 다양한 환경에서 동일 계정으로 접근할 수 있는 통합 저장소이자 폴라리스 오피스 생태계의 중심으로, 문서 동기화 및 공유를 지원한다. 또한 클라우드 저장소의 특성상 접근 기록과 삭제 흔적이 남기 때문에 포렌식 분석 측면에서 중요한 의미를 갖는다. 따라서 본 연구에서는 My Polaris Drive에 초점을 맞추어, 저장된 문서 관련 데이터를 획득하기 위한 방법을 기술한다.

My Polaris Drive 데이터를 수집하려면 먼저 SID 값을 확보한다. 이후 SID를 포함하여 관련 HTTP API를 호출하면 사용자 정보, 최근 사용 파일 목록, 폴더 내 전체 파일 목록, 휴지통 내 파일 목록, 파일 다운로드 결과 등을 응답으로 반환한다. 수집에 활용한 HTTP API 요청을 전달하면 URL은 Table 17.과 같다.

Table 17. Classification of request and response data for data acquisition

URL	Request	Response
https://www.polarisoffice.com/api/1/account/userinfo	"Cookie": ("SID={SID}; ")	User Information
https://www.polarisoffice.com/api/1/drive/recentlistv2		List of recently used files
https://www.polarisoffice.com/api/1/drive/list	"Cookie": ("SID={SID}; ") and fileId (browse within subfolders)	List of all files in a folder
https://www.polarisoffice.com/api/2/drive/download/{File Unique ID}	"Cookie": ("SID={SID}; ")	Download file
https://www.polarisoffice.com/api/1/drive/hiddenbyuseroperationlist	"Cookie": ("SID={SID}; ") and parentId (Id of root folder)	List of files in the Recycle Bin
https://www.polarisoffice.com/api/1/drive/sync	"Cookie": ("SID={SID}; ") and hide: "RELEASE"	Recover files from the Recycle Bin to their original folders

사용자 정보 조회에 사용한 URL은 이메일, 사용자명, 사용자 고유 ID,

가입 일시 등을 응답으로 반환한다. 최근 사용한 파일 목록 URL과 폴더 내 전체 파일 목록 URL에는 파일명, 파일 고유 ID 등의 정보가 포함되어 있다. 이때 최근 사용 파일 목록은 가장 최근에 사용된 파일의 정보만 제한적으로 가져온다. 파일 다운로드 URL은 파일의 고유 ID를 URL 경로에 포함해 요청하며, 응답 본문으로 파일 데이터가 전송된다. 휴지통 URL은 폴라리스 오피스의 휴지통에 보내진 파일 항목 목록을 반환하며, 마지막으로 사용자 정보에 해당하는 URL에 필요한 파라미터를 포함하여 HTTP API 요청을 전달한 결과 응답으로 온 데이터는 Table 18.과 같다.

Table 18. Classification of response data related to user information

Key	Value
adLocale	region
driveUsage	Drive utilization rate
email	User Email
fileCount	Number of files in the drive
fullName	User name
userId	User Unique ID
timeRegist	Registration Date
birthYear	Year of birth
gender	Gender
jobCode	Occupation category
jobDetail	Other Occupations

먼저 adLocale 값은 해당 지역을 의미하며, 한국의 경우 'KR'로 표기된다. driveUsage 값은 현재 드라이브에 저장된 문서의 용량을 나타내며 바이트 단위이다. email은 사용자의 이메일이자 ID이며, fileCount는 드라이브에 저장된 항목 중 폴더를 제외한 파일의 개수를 나타낸다. fullName은 사용자의 이름을 의미하고, userID는 유저의 고유 ID, timeRegist는 가입 날짜를 유닉스 타임스탬프 형식으로 나타낸 값이다. 이후 birthYear부터 jobDetail까지는 userDetailDTO 객체의 하위 속성값들이다. 이 중, birthYear 값은 사용자의 출생 연도를 나타내며 나이를 확인할 수 있다. gender 값은 성별을 나타내며, 'M'은 남성, 'F'는 여성으로 구분된다. jobCode 값은 회원 정보 내 직업을 선택할 수 있는 드롭다운 목록에서 선

택한 사용자의 직업 범주를 의미한다. jobDetail 값은 회원 정보 페이지의 직업 선택 드롭다운에서 '기타'를 선택한 경우, 추가 입력 필드에 사용자가 직접 입력한 값을 의미한다. 전체 파일 목록에 해당하는 URL에 필요한 파라미터를 포함하여 HTTP API 요청을 전달한 결과 응답으로 온 데이터는 Table 19.와 같다.

Table 19. Response data classification related to the full list of files

Key	Value
parentId	ID of parent folder
fileId	Unique ID of the file/folder
fileName	File/folder name
fileType	File type
size	File size
lastModified	Last modified date
fileRevision	Number of revisions
lastModifiedRevision	Last modified version
hide	Whether a file exists in the Recycle Bin
deletedTime	File deletion date
starred	Marked 'Important Document'
shared	Show shared documents
lastAccessTime	Last file access date
path	File path in Polaris drive
userid	Author's User ID

먼저 parentId는 부모 폴더의 ID를 의미하며, 현재 파일이 어떤 폴더 하위에 위치해 있는지 알 수 있다. 단 삭제된 폴더는 모두 루트 폴더의 ID로 고정된다. 이후 fileId부터 userId까지는 directoryInfo 객체의 하위 속성값들이다. 이 중, fileId는 파일 혹은 폴더의 고유 ID를, fileName은 이름을 나타내며, 파일일 경우 '파일명.확장자', 폴더일 경우 '폴더명' 형식이다. 드라이브 최상위 폴더는 'drive'로 표기된다. fileType은 객체의 종류를 나타내며, 일반 파일은 'FILE', 폴더 및 드라이브는 'DIR'로 구분된다. size는 파일의 용량을 나타내며 바이트 단위이고, 폴더일 시 0으로 표시된다. lastModified는 마지막으로 수정한 시간을 유닉스 타임스탬프 형식으로 나타낸다. fileRevision은 파일의 개정 횟수를 나타내는 값으로 1부터

시작하며, 파일의 내용이 편집되었을 때 카운트가 올라간다. lastModifiedRevision은 파일의 마지막 수정된 개정 버전을 나타내는 값이다. 예를 들어 폴라리스의 웹 툴 기능인 문서편집 기능으로 문서를 편집한 후 저장하였을 때 카운트가 올라간다. hide와 deletedTime은 파일을 삭제할 경우 값이 각각 true와 파일을 삭제한 시간을 유닉스 타임스탬프 형식으로 나타낸다. starred는 중요 문서 표시한 파일에만 true로 나타나며, shared는 공유한 문서에 true로 값을 나타낸다. lastAccessTime은 파일에 마지막으로 접근한 시점을 유닉스 타임스탬프 형식으로 기록한다. 파일을 열었더라도 로딩이 완료되지 않은 경우에는 값이 갱신되지 않지만, 로딩이 완료되어 파일의 내용을 확인할 수 있을 때는 해당 값이 갱신된다. path는 폴라리스 드라이브 내 경로를 의미하며, 최상위 폴더의 경로는 'PATH://drive/'이다. 예를 들어 '새 폴더'라는 하위 폴더에 파일이 존재할 경우 경로는 'PATH://drive/새 폴더/'가 된다. 마지막으로 userId는 해당 파일 또는 폴더를 생성한 사용자의 고유 ID를 의미한다.

3) 안티 포렌식 행위에 따른 클라우드 데이터 변화 분석

My Polaris Drive에는 '파일 삭제'와 '영구 삭제'라는 두 가지의 안티 포렌식 기능이 존재한다. 먼저 '파일 삭제'는 삭제하길 원하는 폴더나 파일을 클릭한 후 우클릭해 삭제 버튼을 누르면 휴지통으로 이동되는 기능이다. 휴지통으로 이동된 파일은 hide 값이 true로 변경되고 deletedTime이 0에서 삭제한 시간의 유닉스 타임스탬프값으로 변경된다. 이렇게 이동된 파일이나 폴더는 1년 후에 자동으로 영구삭제가 된다. 따라서 그전에 삭제를 원한다면 '영구 삭제' 기능을 사용해야 한다. '영구 삭제' 기능은 휴지통 내에서 완전히 삭제하고자 하는 폴더나 파일에 우클릭한 후 영구 삭제 버튼을 누를 시 완전히 삭제하는 기능이다. 이렇게 삭제된 파일은 더 이상 HTTP API 요청을 통해 획득할 수 없으며, 서버상에서도 삭제가 된다. 즉 '파일 삭제' 기능만 이용하여 파일이나 폴더를 삭제 할 경우 완전한 삭제가 되지 않기 때문에 해당 기능만 이용한 안티 포렌식을 진행했을

시에는 파일 확보가 가능하다. My Polaris Drive에는 '파일 삭제'와 '영구 삭제'라는 두 가지의 안티 포렌식 기능이 존재한다. 먼저 '파일 삭제'는 삭제하길 원하는 폴더나 파일을 클릭한 후 우클릭해 삭제 버튼을 누르면 휴지통으로 이동되는 기능이다. 휴지통으로 이동된 파일은 hide 값이 true로 변경되고 deletedTime이 0에서 삭제한 시간의 유닉스 타임스탬프값으로 변경된다. 이렇게 이동된 파일이나 폴더는 1년 후에 자동으로 영구삭제가 된다. 따라서 그전에 삭제를 원한다면 '영구 삭제' 기능을 사용해야 한다. '영구 삭제' 기능은 휴지통 내에서 완전히 삭제하고자 하는 폴더나 파일에 우클릭한 후 영구 삭제 버튼을 누를 시 완전히 삭제하는 기능이다. 이렇게 삭제된 파일은 더 이상 HTTP API 요청을 통해 획득할 수 없으며, 서버 상에서도 삭제가 된다. 즉 '파일 삭제' 기능만 이용하여 파일이나 폴더를 삭제 할 경우 완전한 삭제가 되지 않기 때문에 해당 기능만 이용한 안티 포렌식을 진행했을 시에는 파일 확보가 가능하다. 먼저 '휴지통의 파일 목록'데이터를 확보할 수 있는 URL에 HTTP API 요청을 전달해 휴지통 내 항목 정보를 확보한다. 여기서 각 항목의 fileId를 추출한 뒤, '파일 다운로드'용 URL에 fileId를 포함하여 HTTP API 요청을 전달하면 '파일 삭제' 안티 포렌식 행위를 한 파일을 획득할 수 있다. 그러나 휴지통에 폴더가 포함된 경우에는 위 방식만으로는 확보할 수 없다. 휴지통 목록 응답에는 폴더 내부, 즉 삭제된 폴더의 하위 파일 및 폴더 정보가 포함되어 있지 않기 때문에 해당 폴더를 곧바로 '파일 다운로드'로 내려받으면 알맹이 없이 껍데기 폴더만 생성된다. 이런 경우에는 휴지통 내 기능인 '복원' 기능을 통해 원래 위치로 되돌린 뒤, 드라이브에서 해당 폴더의 하위 항목을 조회하고 순차적으로 다운로드해야 실제 데이터를 확보할 수 있다.

4) 클라우드 포렌식 절차 및 검증

폴라리스 오피스의 클라우드 스토리지인 My Polaris Drive의 데이터 수집은 쿠키 기반 인증값인 SID 확보에서 시작된다. 이 값은 웹 환경에서 실행 중인 폴라리스 오피스를 대상으로 웹 브라우저의 개발자 도구를 통

해 직접 추출하거나, 서비스가 동작 중인 환경의 메모리 덤프를 분석하여 확보할 수 있다. 확보된 SID는 이후 사용자 정보, 최근 사용 파일, 전체 폴더 목록 등 다양한 API 요청에 포함되어 데이터 획득의 핵심 매개체로 활용된다. 먼저 사용자 정보의 경우 '사용자 정보' URL에 SID값을 넣어 요청을 보낼 시 획득할 수 있다. 다음으로 폴더 구조 분석의 경우,'폴더 내 전체 파일 목록'에서 얻은 데이터 중 fileType이 DIR인 JSON 항목의 fileId를 추출한다. 이 fileId를 다시 '폴더 내 전체 파일 목록' URL에 SID와 함께 추가하여 요청을 반복하면, 루트 폴더 하위에 존재하는 모든 폴더 및 그 내부 파일 목록까지 단계적으로 확보할 수 있다. 그리고 휴지통 폴더 구조분석은 '휴지통의 파일 목록' URL에 SID와 루트 폴더의 fileId를 추가하여 요청을 보내면, 휴지통에 보관된 파일 목록을 획득할 수 있다. 마찬가지로, 이 응답에서도 fileType이 DIR인 항목의 fileId를 추출하여 다시 '휴지통의 파일 목록' URL에 포함하면, 휴지통 내부의 폴더 트리 전체와 그 안의 파일 목록을 확보할 수 있다. 마지막으로 파일 확보의 경우, 이전 과정을 통해 확보한 모든 fileId를 정리한 뒤, '파일 다운로드' URL의 마지막 부분에 각 fileId를 포함하고 SID를 함께 넣어 요청을 반복 전송하면, My Polaris Drive 내에 존재하는 모든 파일을 실제로 다운로드할 수 있다. 이러한 데이터 획득 절차를 도구를 통해 시각적으로 재구성한 결과 Fig. 19.와 같았다.

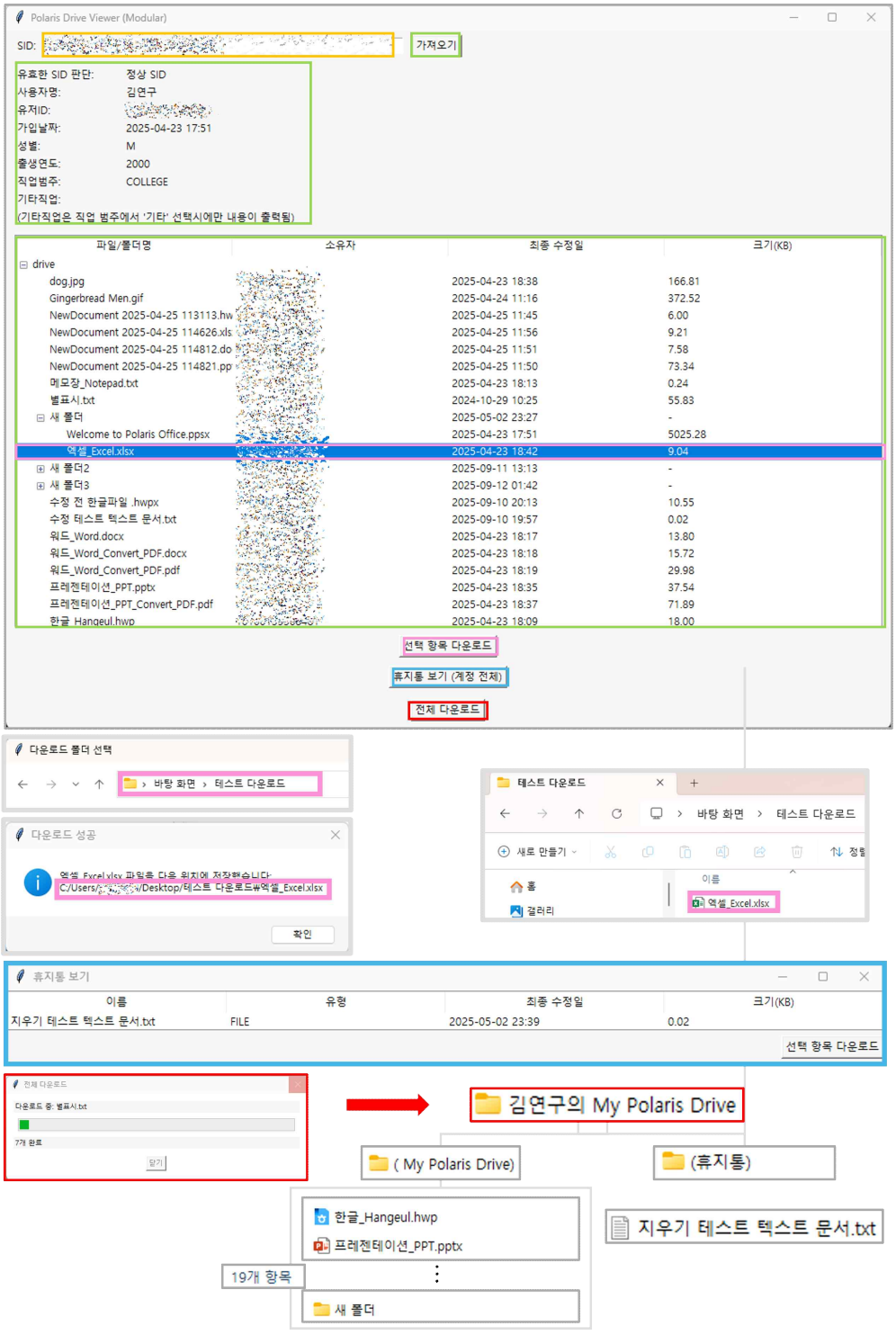

Fig. 19. Reconstructing My Polaris Drive in Polaris Office

이를 통해 My Polaris Drive의 사용자 정보와 파일 구조, 개별 파일까지 직접 확인할 수 있었다. 나아가 파일 다운로드까지 가능함을 검증함으로써, 제안한 절차가 실제 클라우드 환경에서도 충분히 적용 가능하다는 점을 보여주며, 실제 클라우드 데이터 수집을 위해 활용될 수 있는 가능성을 제시한다.

제 6 장 결론

본 연구는 클라우드 서비스의 구조적 특성을 고려하여 설치형과 비 설치형으로 유형을 구분하고, 각 유형에 적합한 데이터 수집 및 분석 절차를 제시함으로써 기존 로컬 기반 포렌식의 한계를 보완하고 서비스 특성에 따른 실질적인 접근 방안을 제안하였다. 설치형 서비스인 잔디의 경우 로컬 아티팩트 분석을 통해 단말에 잔존하는 데이터가 제한적임을 확인하였으며, 이를 보완하기 위해 주요 API 구조를 분석하고 인증 정보를 활용한 서버 기반 데이터 획득 절차를 정립하였다. 이를 통해 클라이언트 측 데이터 확보가 어려운 환경에서도 메시지와 파일 등 핵심 데이터를 재구성할 수 있음을 검증하였다.

비 설치형 서비스인 네이버톡, 디스코드, 폴라리스 오피스에 대해서는 인증 정보 확보 이후 API 호출을 기반으로 클라우드 데이터를 수집하는 절차를 체계화하였으며, 메시지 삭제·숨기기 등 안티 포렌식 기능에 따른 데이터 변화를 분석하여 서비스별 데이터 잔존성을 검토하고 각 서비스별 데이터 획득 도구를 제작해 해당 절차가 유효함을 검증하였다.

본 연구에서 제시한 서비스 유형별 포렌식 절차는 클라우드 기반 메신저 및 협업·오피스 서비스에 대한 수사·분석 단계에서 활용될 수 있으며, 클라우드 환경의 확산에 대응하는 포렌식 표준화 연구의 기반 자료로 기여할 것으로 기대한다.

참 고 문 헌

1. 국내문헌

SAMSUNG SDS. 클라우드 컴퓨팅(Cloud Computing)이란?. https://www.samsungsds.com/kr/cloud-glossary/cloud-computing.html.

신수민, 박은후, 김소람, 김종성. (2020). 디지털 포렌식 관점에서의 Slack 및 Discord 메신저 아티팩트 분석.『한국디지털콘텐츠학회』, 21(4), 799-809.

김영훈, 권태경. (2021). 협업 툴의 사용자 행위별 아티팩트 분석 연구 - 운영환경에 따른 differential forensic 개념을 이용하여.『한국정보보호학회』, 31(3), 353-363.

김한결 위다빈 박명서. (2024). 디지털포렌식 관점에서의 협업 도구 네이버웍스의 데이터 수집 및 분석 연구.『한국정보보호학회』, 34(5), 895-905.

이연주, 김정민, 이성진. (2020). 폴라리스 오피스 포렌식 아티팩트에 관한 연구.『한국디지털포렌식학회』, 14(4), 368-378.

2. 국외문헌

Davis, M. & McInnes, B. & Ahmed, I. (2022). Forensic investigation of instant messaging services on linux OS: Discord and Slack as case studies. *Forensic Science International: Digital Investigation*, 42(3).

Gupta, K. & Varol, C. & Zhou, B. (2023). Digital forensic analysis of discord on google chrome. *Forensic Science International: Digital Investigation*, 44.

Gupta, K. & Lanka, P. & Varol, C. (2024). A holistic digital forensic

analysis of Discord Storage, memory, and network perspectives. *Journal of Forensic Sciences.*

Iqbal, F. & Motyliński, M. & MacDermott, Á. (2021). Discord Server Forensics: Analysis and Extraction of Digital Evidence. *2021 11th IFIP International Conference on New Technologies, Mobility and Security (NTMS)*, 1-8.

Moffitt, K. & Karabiyik, U. & Hutchinson, S. & Yoon,Y . (2021). Discord Forensics: The Logs Keep Growing. *2021 IEEE 11th Annual Computing and Communication Workshop and Conference (CCWC)*, 0993-0999.

Bowling, H. & Seigfried-Spellar, K. & Karabiyik, U. & Rogers, M. (2023). We are meeting on Microsoft Teams: Forensic analysis in Windows, Android, and iOS operating systems. *Journal of Forensic Sciences*, 68(2), 21-26.

Mauricio Piacentini. DB Browser for SQLite_3.12.2. https://sqlite browser.org/.

Maël Hörz. HxD_ 2.5.0.0. https://mh-nexus.de/en/hxd/.

Nirsoft. ChromeCacheView_v2.46. https://www.nirsoft.net/utils/chrome_ cache_view.html.

cclgroupltd. Chromium_dump_local_storage.py. https://github.com/cclgro upltd/ccl_chrome_indexeddb/blob/master/Chromium_dump_local_sto rage.py.

Telerik AD. Fiddler_v5.0.20211.51073. https://www.telerik.com/fiddler.

google. winPmem_2.0.1. https://github.com/Velocidex/WinPmem.

google. Chrome. https://www.google.com/chrome/.

Telerik AD. Fiddler Everywhere_v7.2.0. https://www.telerik.com /fiddler.

ABSTRACT

A Study on data collection and analysis techniques on cloud-based platforms

We, Dabin

Major in Convergence Security

Dept. of Convergence Security

The Graduate School

Hansung University

This study categorized cloud services into installed and non-installed types and proposed data collection and analysis procedures appropriate for each type. As JANDI, an installed service, exhibits limited persistence of local artifacts, we first analyzed local artifacts such as the database and cache. We then identified key APIs and developed a procedure to directly acquire server data by reconstructing API requests using authentication information obtained from memory. We then constructed a hypothetical scenario to apply the procedure and reconstructed an actual chat room to verify the validity of the analysis process. Non-installed services such as Naver Talk, Discord, and Polaris Office operate based on web browsers, leaving little meaningful data locally. Therefore, this study first acquired credentials and used them to make API calls to collect server data, and then analyzed the collected data. We also compared data changes resulting from anti-forensic functions such as message deletion, hiding, and file deletion to examine data persistence across services. Finally, we developed data collection tools specific to each service and verified the validity of the procedures in a real-world

environment. The results of this study are expected to provide a useful example for establishing digital forensic procedures for cloud-based collaboration tools and messengers, contributing to the establishment of a standardized analysis methodology for investigations in the growing cloud environment.

【Key words】 Digital Forensics, Cloud Forensics, JANDI, Naver Talk, Discord, Polaris Office

본 논문의 제4장(설치형 클라우드 서비스)은 학위논문 연구과정에서의 연구결과를 다음의 학술지에 게재한 바있으며, 주된 내용의 일부는 하기의 발표 논문내용과 같은 내용이며, 한성대학교 대학원 학술지 게재논문의 인용 지침을 준수하였습니다.

“위다빈, 김한결, 박명서. (2024). 윈도우 환경에서의 협업 도구 잔디 아티팩트 수집 및 분석 연구. 『정보보호학회논문지』, 34권(5).”