Graduation Project 2 Final Report

Forensic Analysis of Ephemeral Messaging:

A Multi-Vector Snapchat Forensic Study

By:

Austin Webber & Kaiden Koran

British Columbia Institute of Technology
Digital Forensics and Cybersecurity Program
March 2026

Table of Contents

Abstract

This project investigated the recoverability of Snapchat's ephemeral messaging artifacts through three vectors: Android device forensics, iOS device forensics, and interception of network traffic. A controlled laboratory environment was created - with a Linux-based software access point running mitmproxy and Frida for interception of encrypted network traffic. Four devices were set up with Snapchat test accounts, including: A rooted Samsung Galaxy A8, a non-rooted Samsung Galaxy S8 (alongside a subsequent full file system extraction with Cellebrite), an iPhone 13, and a jailbroken iPhone XS. Intercepted network traffic was captured through a secondary avenue using tcpdump for Wireshark analysis.

Network dump analysis confirmed the continued presence of QUIC traffic despite the interception setup - preventing recovery of content from packets. Metadata related to DNS queries, server addresses, timing patterns, and server contact were successfully recovered. Android devices with root or full file system access provided a significant amount of evidence, including extensive artifacts from arroyo.db and main.db. This included cached media, chat messages, stories, and voice notes. iOS encrypted iTunes backups provided metadata for conversations through the InteractionC databases and keychain authentication artifacts, but direct message content was unrecoverable without full file system acquisition.

These findings confirm that all types of ephemeral data still persist on device storage despite the perceived non-recoverability from Snapchat's disappearing message function. Recovery success depended greatly on device access level, operating system, and acquisition type. This research contributes to forensic literature on Snapchat by comparing recovery outcomes across multiple Android and iOS device models.

Introduction

Over the past two decades, social media has evolved into the most widely used method of online communication. Platforms like Instagram and Snapchat allow users to share personal experiences through various forms of media, such as photos and messages. Snapchat distinguishes itself by formulating its core design around the concept of ephemerality - in which, content is intended to disappear after viewing. This function has made Snapchat popular for users who think about privacy and data, but it has also created obstacles for forensic investigators who rely on digital evidence to build their case.

Previous research has shown artifacts persist to different degrees depending on the operating system, device access, and the extraction methods employed. Studies from ScienceDirect (Alyahya & Kausar, 2017) and Canterbury University (Azhar et al., 2020) have documented recoverable databases and cached media on Android devices. iOS research has reinforced the more restrictive nature of Apple's security model (Bates, n.d.). Few studies directly compare recovery success on both platforms with controlled conditions - and even fewer have examined the implications of Snapchat's usage of the QUIC protocol and TLS 1.3 encryption.

It is up to investigators to adapt to new methods for recovering or analyzing leftover data while avoiding the violation of any legal and ethical standards (LawShun, 2025). With content designed to self-destruct after a set time, standard forensic methods that rely on persistent storage may be ineffective. Investigators should consider multiple acquisition points, including network, cloud, and device level artifacts to reconstruct communications meaningfully. With Snapchat's design any delays between incident and examination can result in the permanent loss of evidence - which makes acquisition an urgent matter.

The main research question guiding this project is: What data can be recovered in an investigation where Snapchat is the primary source of evidence? This was explored through three avenues: network traffic interception and analysis, physical and logical forensic extraction from Android devices, and encrypted iTunes backup analysis from iPhones. Test accounts were set up for each device, and a pre-determined set of Snapchat activities were performed under controlled conditions. This helped establish a known baseline for comparison of recovered artifacts.

The goal of this study was to determine whether network traffic from Snapchat could provide insight into user activity - and whether local device artifacts could be recovered after supposed deletion. This research was conducted entirely within a controlled laboratory environment with test accounts, and no third party or authentic user data was utilized. The study was designed for compliance with the ACPO principles for digital handling of evidence, and Canada's PIPEDA guidelines.

Literature Review

The Rise of Ephemeral Messaging in Social Media

Platforms including Instagram, Facebook, and Snapchat have drastically altered how people interact online. In July of 2011, three Stanford students introduced the concept of self-destructing or "ephemeral" messaging through the app Picaboo. This initial app concept failed to gain traction, but the applications relaunch under the name “Snapchat” eventually grew popular amongst young adults (*Snapchat | History Timeline*, n.d.). With this rise in popularity, "ephemeral" messaging became widely adopted as a communication standard - largely due to its assurance of privacy and data control. Today, Snapchat is recognized as one of the largest platforms that uses this feature - but other apps such as Signal, Wickr, and Telegram now utilize it as well. (*Azhar & Chamberlain, 2020; IARAS, 2025*)

The introduction of disappearing content has redefined the standard for users’ expectations of privacy, though technical implementations and retention policies differ across services (*CLRN, 2025; TechCrunch, 2024*). This feature created many new challenges for forensic and cybersecurity investigators. The way ephemeral messages are designed makes tracing evidence, verifying communications, and retrieving data deleted during investigations difficult (*DIVA Portal, 2023*). While these platforms continue to improve their encryption and temporary data practices, forensic research suggests that ephemeral content can still leave recoverable traces behind. (*Alyahya & Kausar, 2017; Aji et al., 2017*). Findings from this study support this claim for both Android and iOS devices, since artifacts were successfully recovered from the device storage even though content was viewed and pending deletion.

The first things that come to mind when one hears the term “self-destructing data,” may typically be messages, videos, and pictures. Realistically, that is only a part of the bigger picture. More valuable evidence may be found inside the device memory, storage cache, or even in the network traffic during the communication. In terms of temporary data, not all data is actually temporary and gone forever. Even if it was made to disappear, it can leave traces of temporary files and metadata on the device itself. Some of it will always be kept or sent to a certain place on a mobile device (*Azhar & Chamberlain, 2020; Alyahya & Kausar, 2017)*. This study confirmed that system images from both Android and iOS devices retain Snapchat artifacts that are still recoverable. Artifacts ranged from full message contents on a rooted Android device to metadata and authentication tokens on iOS.

Legal and Ethical Considerations

It is critical to acknowledge the legal and ethical implications of this project. In order to attempt ephemeral message recovery - a controlled lab will be set up. This will include a laptop, an iPhone, and an Android phone. The data analyzed will solely originate from test accounts created on each device. All

experiments will be carried out on an isolated laboratory network to avoid inadvertent collection of external traffic.

There will be no third-party or external user data utilized for the purpose of this project. Thus, no breach of privacy or violations of terms of service for Snapchat's app usage agreement will occur. The project recognizes the potential ethical concerns associated with inspection of user data, and aims to meet the forensic research standards stated in the ACPO principles (*AthenaForensics, 2023; ForensicControl, 2025*). This project intends to stay in compliance with legal consideration denoted under Canada's PIPEDA data protection framework. No attempt will be made to access Snapchat's servers (*Office of the Privacy Commissioner of Canada, 2024*).

Traffic interception will not occur on non-research devices or accounts, and vulnerabilities will not be exploited on systems beyond our test devices. The intention is for this project to be used only to bolster technical understanding and provide educational support. Device models, OS, app, and toolchain versions will be reported to support ethical review.

The Cultural Psychology of Ephemerality

Studies focusing on how people interact with ephemeral messaging platforms hints at the appeal going beyond simple privacy. A qualitative study from 2016 found that Snapchat users felt gratification for not only the privacy it afforded, but also from opportunities created with short-form and disappearing messages (*Waddell, 2016*). Users felt less anxious about presenting themselves, due to the content's short life span - opening the door for more spontaneity and creativity. The modality through which communications are exchanged - namely, short form videos and photos - made users feel they could engage in more nuanced self-expression than over text. On top of providing a way for users to express themselves, many of the respondents said Snapchat was a primary method of staying in touch with long-distance family and friends. The interviewed users showed moderate trust in the privacy of temporary content. However, many respondents nonetheless displayed skepticism towards the content having actually "disappeared," believing it may still persist in some form (*Waddell*). Results from this study validated that skepticism, since actionable traces of artifacts were identified and recoverable between Android and iOS devices tested.

Snapchat's streaks feature encourages users to engage in interaction with others. This unique function requires interaction between two users within 24 hours in order to maintain and extend the streak. Problematic mobile device usage was associated with this feature due to the FOMO (Fear of Missing Out) that it induces (*Essen & Ouytsel, 2023*). Using experience sampling and interviews of 154 college students, *Bayer et al. (2015*) observed that Snapchat interactions are seen as more engaging and mood-enhancing versus older forms of communication. Participants reported that because content disappears - they felt more present, and more willing to share varietal content versus the curated, idealized content typically seen on platforms like Facebook. However, Snapchat interactions were connected to lower amounts of social support than through other communication channels.

The MAIN (Modality-Agency-Interactivity-Navigability) model proposed by Sundar (2008) may assist in explaining why Snapchat and similar ephemeral apps have become so popular. The modality (rich media), sense of agency (deletion timers), interactivity (real time and near real time exchanges), and navigability (ease of use) all suggest why users adopt ephemeral messaging (*Waddell, 2016*).

Statistical Usage Data and Cybercrime/Forensics Relevance

Several empirical studies provide data on how ephemeral messaging is used - and what users expect from platforms with disappearing content. A survey of 1,515 Snapchat users found that decisions surrounding what content they shared was primarily influenced by the intended audience and sensitivity of potential content (*Habib et al., 2019*). Public snaps more commonly contained general experiences; private snaps were more likely to include personal or sensitive content. This suggests that forensic investigations of privately shared content may be more relevant than publicly posted stories. The results of this study tested those protocols and included both private messages and stories to test the recoverability of that content.
Bayer et al. (2015) found that daily Snapchat usage was frequent amongst college students (simple, spontaneous content), and that ephemeral communication was preferred over more permanent options for everyday conversations. In "The Allure of Privacy..." Waddel (2016) found that while surveyed respondents denied using Snapchat for the sharing of risque photos and videos, many assumed their friends were utilizing ephemeral messages for this purpose. The "third person effect" was applied to this study, and it was seen that replies echoed similar observed perceptual bias seen in The Third Person Effect Review and Synthesis - where this form of communication was assumed by respondents to be more likely to incite questionable and explicit behaviours in friends than in themselves. (*Perloff, 2009*)

Implications for Forensic Investigations

The forensic approach is everchanging. Insight from studies into the psychology of users who utilize ephemeral applications may prove useful for future forensic strategy. While many users are aware of, or suspect, traces left behind after ephemeral communication, many others are not. But those who may suspect traces do not know what exactly is left behind. Forensic investigators should anticipate artifacts where users do not (*Mutawa et al., 2016; Heath et al., 2023*).

The majority of ephemeral communications are not typically useful as direct evidence due to their typically unremarkable nature (*Habib et al., 2019*). While they may not offer irrefutable evidence of wrongdoing on their own, they may still provide context clues (timestamps, locations) and assist in establishing suspect's behavioural patterns. Since ephemeral features only allow content to briefly exist on a visible and easily accessible level, forensic investigators must minimize the delay between incident occurrence and data acquisition (*Heath et al., 2023*). This study was able to confirm this occurrence, because all device data was acquired right after the controlled communication happened.

User behaviour alone provides investigative signals. Many users share in "small moments," with reduced anxiety surrounding self-presentation (per "Sharing the Small Moments") (*Bayer et al., 2015)*. This

proves opportune for indirect but still relevant evidence collection by investigators though timestamps, viewer lists, and thumbnails. Forensic assumptions that "only sensitive, or obscene content matters" may forgo the necessary depth of investigation and prevent securing a conviction. (*Bayer et al., 2016; Mutawa et al., 2016*)

Encryption and Network Security on Snapchat

One of the more important aspects that comes to mind is the encryption and security measures that are applied to the data. Snapchat is a "mixed bag" that does not have full end-to-end encryption for all message types (*CLRN, 2025*). The data going from the client Snap's servers uses transport layer encryption as well as the data stored on the servers. However, chats on the app are not encrypted by default. This shows that there is still a possibility that data from chats are still recoverable, considering the lack of encryption or security used on the chats themselves (*CLRN, 2025*). An article from TechCrunch reported that Facebook had secretly been exploiting the lack of encryption for a competitive advantage in a project they called Ghostbusters (*TechCrunch*, 2024).

In 2016, Facebook created a secret project called Ghostbusters that was meant to intercept and decrypt Snapchat's traffic. The goal was to investigate how users were interacting with their competitors. It was eventually discovered that Facebook had been paying teenagers to use a VPN service called Onavo which was used to spy on the user's web activity. This was later shut down after TechCrunch exposed what was happening (*TechCrunch*, 2019).

Snapchat's method of transport encryption (QUIC and TLS 1.3) has evolved, alongside application level protections like certificate pinning (*Snap Engineering, 2021*). Certificate pinning – accepting only authorized or pinned certificates for a secure session – prevents the decryption of packets that are intercepted when using tools like Wireshark or MITMproxy (*GitHub TLS-Bypass, 2024; Barcaroli, 2023*). The network analysis performed in this study confirmed that these protections were used. Even though Frida was deployed to bypass certificate pinning and mitmproxy for traffic interception, frames with QUIC payload protection were still very vague in content analysis.

QUIC moves TLS into the transport layer and encrypts a larger portion of the handshake and transport. Mitmproxy has historically not supported QUIC, as it is a relatively new transport protocol. It is only within the last five years that the developers began work to support it. Release notes from August in 2023 denote the release of mitmproxy 10 - which included experimental support for QUIC and HTTP/3 reverse proxies (*Hils, 2023*). The next version - mitmproxy 11 - introduced full support for HTTP/3 in both transparent and reverse proxy modes. It also added support for TLS 1.3 Post Handshake Authentication (*Jain & Hils, 2024*). The second caveat is certificate pinning. Certificate pinning is the process of associating a host with a specific expected X.509 certificate or public key. In essence - on first encounter, a certificate or public key is associated (pinned) to the specified host. The process of pinning at this stage is called key continuity. This can be circumvented by an attacker by establishing a privileged position before this encounter occurs (*LRQA, 2018).* These findings all indicate that Snapchat's encryption and

network security techniques greatly limit the possibility of externally observing or capturing user data. Results from network analysis performed directly confirmed this.

Forensic Recovery and Data Extraction

When attempting to recover data from mobile phones, one of the first places to check is the system files themselves. As there are several places for data to be stored, it can prove difficult to know where to start looking. Particularly as Android and iOS devices have largely differing methods in system/file design, encryption techniques, and security methods (Azhar et al., 2020). For Android devices, articles suggest that most data may be recoverable through Snapchat's install folder location. ScienceDirect and Canterbury recovered artifacts found in Snapchat's main archive /data/com.snapchat.android/, which contains folders for cache, profile data, snaps, stories, received photos and videos, and databases such as tcspahn.db and db-journal (*Alyahya & Kausar, 2017; Aji et al., 2017*). Results from this study confirmed that Snapchat artifacts are located under /data/data/com.snapchat.android/, where databases arroyo.db and others were found to confirm conversation IDs. The following are the contents of each artifact.

(*Alyahya & Kausar, 2017; Aji et al., 2017*)

Artifact	**Contents**
/data/com.snapchat.android/	Main Snapchat archive in app's folder.
Cache folder in main archive	Chats, profile data, snaps, stories, and received images and videos.
Databases folder in main archive	Databases like tcspahn.db and db-journal.
Snapchat folder in main archive	Snaps containing images and videos that were sent and/or deleted.

iOS devices are well known to present even greater challenges when it comes to file system encryption and sandboxing. Cellebrite Physical Analyzer is capable of extracting valuable yet limited artifacts and metadata. This may include timestamps for messages, user IDs, and media references. The following are acquired artifacts that pose as possible pieces of evidence using Magnet AXIOM and Cellebrite Physical Analyzer (Bates, n.d.):

Magnet AXIOM

Artifact	**Content**
FullFileSystem.1.dar\private\var\mobile\Container	Possible location of where Snapchat archive is

s\Data\Application\<AppID>\	located.
\Documents\gallery_data_object\ in main archive	Messages, photos, videos, stories, and memories.
\Library\Caches\SCPersistentMedia\	Chat videos

Table A - iPhone artifacts found by SiennaBates on MagnetAXIOM

Cellebrite

Artifact	**Content**
DarArchive\root\private\var\mobile\Containers\Data\Application\<AppID>\	Potentially contains messages, photos, videos, stories, and memories
\Documents\gallery_data_object\ in main archive	Messages, photos, videos, stories, and memories
\Library\Caches\SCPersistentMedia\	Chat videos

Table B - iPhone artifacts found by SiennaBates on Cellebrite Physical Analyzer

These tools are capable of retrieving some iOS artifacts, but information is relatively limited. There are data recovery tools available to the general public, from Alphr.com and AnyRecover. These are known for being able to restore some deleted Snapchat content (*Johnson, 2023; Doris, 2025*). However, their results are also inconsistent and lack forensic reliability. Collectively, the evidence reviewed suggests that Snapchat's feature of ephemeral messaging deletion works relatively well, but it can not guarantee complete removal. The app will leave traces of data, but the results will differ depending on the device at hand (*Malley, 2021; Rashid & Mastorakis, n.d*). The study's iOS analysis proved that encrypted iTunes backups contained conversation metadata and authentication artifacts; but without a full file system extraction, direct messages were not recovered.

Research Gaps

Although tools used throughout this study are very effective at recovering actionable evidence, many limitations still exist. While there are working solutions for analyzing network traffic, very few tools can properly inspect data transmitted over newer protocols like QUIC and TLS 1.3 (*QUIC at Snapchat - Snap Engineering*, 2021). Most existing research still focuses on TLS 1.2 and older HTTP/TCP connections, meaning the forensic community has yet to fully catch up with modern encryption standards. After analysis of network captures from the controlled lab, this limitation was immediately noticed due to payloads encrypted by Snapchat's QUIC protocol. These could not be decrypted even though a certificate pinning bypass was successful.

There are also challenges related to the differences between operating systems. iOS and Android handle files and system access in very different ways, which affects what investigators can recover. The Android devices are much more accessible for forensic extraction compared to iOS, which are much more encrypted and secure. Few studies directly compare recovery success rates between the two platforms

under identical conditions, leaving it murky as to whether ephemeral data behaves the same way across both environments. The purpose of the study was also to compare the differences in analysis of both mobile operating systems. This was done by performing the same actions and analysis on both devices with identical account settings and controlled variables.

Finally, some certificate-pinning bypass methods exist, but their legal and procedural validity in forensic contexts is still uncertain. Research is noticeably lacking compared to research in other forensic areas (*Cyberly, n.d.*). This project ensured to stay within the standard and legal compliances for forensic practices, and is aware of the grey areas related to techniques like Android rooting.

Experimental Methods

The devices used in this study are summarized in Table 1, with forensic workstation specifications in Table 2 and all software tools and versions in Table 3.

Table 1. Hardware specifications for devices used in the study

Device	OS Version	Role	Snapchat Version
Samsung Galaxy A8	Android 9 / One UI 1.0	Rooted Android (Ryoma Echizen)	Latest at time of test
Samsung Galaxy S8 (SM-G950W)	Android (stock)	Non-rooted Android (John Titor)	Latest at time of test
iPhone 13 (iPhone14,5)	iOS 16.1 (Build 23B85)	iOS test device (Dean Winchester)	13.79.1
iPhone XS (iPhone11,2)	iOS 15.4.1 (Build 19E258)	Jailbroken iOS (Sam Winchester)	13.75.0

Table 2. Forensic workstation and network capture hardware

Component	Specification
Forensic Laptop (Network Capture)	Lenovo ThinkPad T480s
Processor	Intel Core i7-8650OU
Graphics	Intel UHD Graphics 620
RAM	16 GB
Storage	4 TB SSD
Wireless Adapter	Intel (2.4 GHz STA-AP mode)

Table 3. Software tools and versions used in the study

Tool	Version	Purpose

Magnet AXIOM Process/Examine	9.8.0.46347	Primary forensic analysis platform
Cellebrite Inseyets (PA)	10.9.0.3029	Secondary forensic analysis platform
DB Browser for SQLite	v3.13.1	Manual database examination
FTK Imager	4.5.0.3	Lx01 image extraction
Wireshark	Current at time of analysis	PCAP analysis and protocol inspection
mitmproxy	12.2.1	Transparent HTTPS proxy
Frida	17.6.0	Runtime instrumentation / SSL bypass
Odin3	v3.14.4	Samsung firmware flash
Magisk	v30.6	Android rooting / root concealment
Frija Tool	v2.0.23364.3	Samsung firmware download
rootAVD	Current at time of use	Android virtual device rooting
hostapd	System package	Software access point creation
dnsmasq	System package	DHCP for isolated network
tcpdump	System package	Raw packet capture
Dopamine (via TrollStore)	Current at time of use	iOS 15 semi-untethered jailbreak
GrayKey	Current (mentor-assisted)	Advanced mobile acquisition
Cellebrite UFED Premium	Current (mentor-assisted)	Full file system extraction

Research Methods

Multiple methods were used in order to determine the success rate of recovering ephemeral Snapchat data through different forensic acquisition tools. The qualitative attributes of the study were the device artifact types and locations, and included quantitative elements from the network frame analysis, and logical data and artifact capacities. Three main areas of acquisition vectors were investigated throughout this project: network traffic data, Android device forensic analysis, and iOS forensic analysis. Each of these areas were analyzed independently and then cross-examined to verify found evidence.

The study followed a controlled lab setup. Multiple test accounts were created to be used on multiple different Android and iOS devices. These accounts all used predetermined credentials, along with a basic set of actions that were performed to set a standard baseline for data analysis. All mobile devices were connected to an isolated network setup by a Linux laptop with network traffic being monitored. This

design ensured that all device traffic being sent was seen by the controlled network infrastructure. This allowed for network and device data collection all at the same time when it was being acquired.

The methodology of the study was set up in such a way that an independent researcher could reproduce the scenario with equivalent equipment. The device models, operating systems, program and application versions, and forensic tool versions were all documented. The actual forensic acquisition process followed standard digital evidence handling rules, which includes hash verification of the data extracted to maintain integrity.

Data Gathering Tools

Network Traffic Capture Tools

Network traffic was captured with the use of several tools on the Linux laptop, including: hostapd, dnsmasq, mitmproxy, Frida, and tcpdump. Hostapd was configured so the laptop's wireless adapter could act as a software access point (AP) - this operated on the 2.4 Ghz band due to the Intel adapter's STA-AP mode hardware limitations. Dnsmasq provided DHCP and DNS services for the connected devices. Iptables handled NAT translation and packet forwarding between the software AP (ap0) and the interface facing the internet (wlp61s0.) This infrastructure was constructed as a means of routing all device traffic through the laptop, allowing for active and passive interception.

Mitmproxy was deployed as a transparent proxy for interception of HTTPS traffic. A root certificate authority (CA) certificate was generated and installed on all test devices, with the intent to allow the proxy to decrypt TLS-encrypted communications. Frida, a dynamic instrumentation toolkit, was installed and setup on two devices: the Samsung Galaxy A8 and the iPhone XS. A publicly available SSL unpinning script was executed on the A8 via Frida, with the intent to bypass Snapchat's certificate pinning mechanism. Use of the script was attempted with the iPhone XS but was unsuccessful, as the script specifically targeted Android Snapchat deployments. Frida was not deployed on the iPhone 13 or Samsung Galaxy S8 - the iPhone lacked a publicly available jailbreak, and the S8 was unable to be rooted. Tcpdump was used for the capture of packet data on ap0 and wlp61s0, to produce PCAP files that could be subsequently analyzed in Wireshark.

Several limitations affected the network capture's efficacy. While iptables rules were configured to drop UDP port 443 traffic (forcing Snapchat to fall back from QUIC to TCP, to allow for mitmproxy to intercept), these rules were unsuccessful. The SSL unpinning script failed to correctly hook the certificate pinning mechanism, and the A8 received certificate rejections from Snapchat servers when attempting to connect through mitm's transparent proxy. As a result, the majority of captured application traffic used

QUIC - and mitmproxy was largely ineffective as a capture tool. The transparent proxy only captured plaintext HTTP traffic, capturing connectivity checks for devices: Apple's captive portal probes (captive.apple.com), and Android's (google.com/generate_204).

Android Acquisition Tools

Several tools were used in order to acquire actionable Android artifacts as evidence. However, these tools were dependent on the device's access level, and the type of extraction performed. The Samsung Galaxy A8 was rooted using Magisk through the Odin3 firmware flashing tool, which allowed for a full file system extraction using Android Debug Bridge (ADB) shell commands. Magisk was further configured with Zygisk and DenyList to properly bypass administrator escalation detection on Snapchat. The adb shell was utilized to generate a tar archive of the /data partition, and was pulled onto a forensic workstation for further analysis. Magnet AXIOM process and Examine (versions documented and noted at time of acquisition) were the primary tools which provided the most evidence. Magnet AXIOM Examine made the analysis process much easier to analyze specifically for Snapchat-related artifacts. DB Browser for SQLite (v3.13.1) was used to manually confirm database files, specifically arroyo.db and main.db.

On the Samsung Galaxy S8, this device had difficulties with attempts to successfully root the device. This was because the device is a Canadian variant using a Snapdragon chip which has the OEM unlocking option (required for rooting) unavailable in Developer Options. An attempt for a full logical extraction was made through the Magnet AXIOM process, however it kept resulting in insufficient artifacts related to Snapchat. Instead, with the assistance of the project's RCMP mentor, a full file system extraction was obtained through Magnet GrayKey and Cellebrite UFED Premium. This provided an Lx01 image that was extracted through FTK 7.3 and then processed in Magnet AXIOM Examine for Snapchat-specific artifact analysis.

iOS Acquisition Tools

iOS device acquisition was conducted through encrypted iTunes backups created on a MacBook Air. The iPhone 13 and the iPhone XS were connected to iTunes, encryption passwords were created, and full backups were completed. These encrypted files were then transferred to the primary forensic workstation and fed to Magnet AXIOM Process and Cellebrite Inseyets for analysis with the encryption password.

Using both platforms provided an alternative view of the artifact data for comparison and contrast. The combination allowed for cross-validation of identified relevant artifacts through completely independent forensic platforms.

A semi-untethered jailbreak was performed on the iPhone XS via Dopamine, installed through TrollStore. The jailbreak was required to install the Frida server on the device to attempt bypassing certificate

pinning during the network capture. The jailbreak was not utilized to attempt the extraction of physical data.

Additional Analysis Tools

In order to fully analyze the traffic going in and out of the AP on the Linux laptop, Wireshark was used to analyze the PCAP files generated by tcpdump. This software allowed for protocol-level inspection of captured network data, including DNS queries, TLS handshakes, and QUIC protected payloads.

Odin3 (v3.14.4) was used specifically for the successful Samsung Galaxy A8 rooting process. Using Frija Tool (v2.0.23364.3), it generated stock BL, CP, and CSC firmware files. After generating the stock firmware, a custom Magisk-patched AP firmware file was required for the root to be successful in Odin3.

Describe the Data

Data collected throughout this study consisted of three primary categories: network traffic captures, Android file system extractions, and iOS encrypted iTunes backups. Each category had their respective data types, formats, and variables that were relevant to the forensic analysis of Snapchat.

Network traffic data was stored using the PCAP (packet capture) format, and generated by tcpdump during the controlled period of communication. Two PCAP files were collected: one from the AP interface (ap0), capturing traffic exchanged between devices and the gateway on the laptop. One from the internet-connected interface (wlp61s0) capturing traffic between the gateway and Snapchat's servers. Key information from the network data included source and destination IPs, DNS queries, protocol types (QUIC, DNS, TLS), packet byte payload size, and timestamps.

Android device data was extracted in tar format from the rooted Galaxy A8 (adb commands and Magnet AXIOM) and in Lx01 format from the Galaxy S8 (GrayKey and Cellebrite). Crucial evidence found included Snapchat SQLite databases (arroyo.db and main.db), cached media files in JPEG and MP4 formats, log files (account.txt, usage_stats.txt, battery_stats.txt), and application configuration files. Data recovered contained both qualitative and quantitative elements. Qualitative being message content and display names, quantitative being timestamps, file sizes and conversation and messages identifiers.

iOS data was stored in the encrypted iTunes backup format, which was decrypted by Axiom and Cellebrite's processing with the backup password. Key evidence included InteractionC database records, keychain entries (SCOneTapLoginKeychainKey, fideliusTransferableDeviceGraph, fideliusTransferableIdentityBackup), and EXIF metadata from saved photographs (GPS coordinates, application metadata.) The iOS data was mainly qualitative - primarily sourced from metadata, timestamps and authentication tokens rather than direct evidence of message contents.

Study Conduct

Phase 1: Environment Preparation

The study started with creating the Snapchat test accounts. Four email accounts were created through Proton Mail, and their respective Snapchat accounts were registered on four devices: Samsung Galaxy A8 (Ryoma Echizen), Samsung Galaxy S8 (John Titor), iPhone 13 (Dean Winchester), and iPhone XS (Sam Winchester). A distinct identity was assigned to each device in order to properly track the inter-device communications during analysis (see Table 4).

Assigned Identities (Snapchat, Proton) to all four devices.

Table 4. Test accounts and identities across four devices.

Device	Assigned Identity
iPhone XS	Sam
iPhone 13	Dean
Samsung S8	John
Samsung Galaxy A8	Ryoma

Rooting the Samsung Galaxy A8 required multiple steps. First, Developer Options was enabled by tapping the Build Number seven times in the device's About settings. OEM was then enabled along with USB Debugging within Developer Options. Next, the device was booted into Download mode in order to verify the bootloader was unlocked. Official Samsung firmware was generated using Frija Tool (v2.0.23364.3), and the AP file was patched using Magisk (v30.6) to create a custom firmware (CFW) image. Lastly, Odin3 (v3.14.4) was used to flash the BL, CP, CSC, and Magisk-patched AP firmware files to the device. Once the flash completed, the device rebooted and root access was verified using Root Checker from the Google Play Store and the Magisk application (see Figure A1 and A2).

Due to Snapchat's elevated privilege detection, a root detection bypass was configured on the Galaxy A8 through Magisk's Zygisk module and DenyList feature. Zygisk was enabled through Magisk settings, and "com.snapchat.android" was added to the DenyList (see Figure A3). This prevents Snapchat from detecting the device is rooted. Because the application actively checks for root access, it will not operate or function properly on rooted devices.

An attempt was made to root the Samsung Galaxy S8. But because it was the Canadian variant using a Snapdragon chip, OEM Unlocking was not available in Developer settings. This prevented unlocking the bootloader to continue with the root process. Alternatively, the S8 was used under expected user conditions to simulate what a real forensic evidence seizure would involve.

Phase 2: Network Capture Infrastructure

The network capture infrastructure was designed and implemented on an Ubuntu laptop (Thinkpad T480s, Intel i7, 16GB ram). An initial attempt was made to use an Apple Airport Express as the dedicated access point, but this was abandoned in favour of a software-based access point using the laptop's wireless adapter. Hostapd was installed and configured to initialize an access point named "Snap Forensics" on channel 11 of the 2.4GHz band (See Figure B1). Operation using 2.4GHz was necessary for operation, as the wireless adapter did not support 5Ghz in STA-AP (simultaneous station and access point) mode.

Dnsmasq was configured to provide DHCP services to devices connecting to the software AP (Figure B2). The configuration designated assigned IP addresses should be within the 192.168.100.10-100 range, with a 24 hour lease period. The laptop's AP interface address (static - 192.168.100.1) was advertised as the default gateway, directing all device traffic through the laptop. DNS resolution used Google's resolvers (8.8.8.8, and 8.8.4.4), and DHCP logging was enabled for auditability purposes.

IP forwarding was enabled permanently through the sysctl configuration file, and a bash script was written for iptables rules. The iptables script flushed current rules, set default ACCEPT policies, and enabled NAT translation for traffic leaving the internet-facing interface. To attempt a forced fall back from QUIC, DROP rules were created to block UDP traffic on ports 80 and 443. PREROUTING REDIRECT rules directed all TCP traffic on ports 80 and 443 to mitmproxy's listening port 8080 for interception (See Figure 1).

```
# Rule flush
iptables -F
iptables -t nat -F
iptables -t mangle -F


# Default policies
iptables -P INPUT ACCEPT
iptables -P FORWARD ACCEPT
iptables -P OUTPUT ACCEPT


# NAT
iptables -t nat -A POSTROUTING -o wlp61s0 -j MASQUERADE


# forwarding between ap0 and wlp61s0
iptables -A FORWARD -i ap0 -o wlp61s0 -j ACCEPT
iptables -A FORWARD -i wlp61s0 -o ap0 -m state --state RELATED,ESTABLISHED -j ACCEPT


# Block QUIC (forcing HTTPS fallback)
iptables -A FORWARD -i ap0 -p udp --dport 443 -j DROP
iptables -A FORWARD -i ap0 -p udp --dport 80 -j DROP


# Redirect HTTP/HTTPS to mitmproxy (port 8080)
iptables -t nat -A PREROUTING -i ap0 -p tcp --dport 80 -j REDIRECT --to-port 8080
iptables -t nat -A PREROUTING -i ap0 -p tcp --dport 443 -j REDIRECT --to-port 8080
```

Figure 1. Iptables rules for connectivity, traffic redirection, and network forwarding between ap0 and wlp61s0.

An issue was encountered during initial setup where devices connected to the AP were unable to reach the internet. Troubleshooting revealed that a VPN running on the laptop was interfering with packet routing -

and after disabling it - traffic successfully flowed through the AP interface, NAT, and the internet-facing interface to the internet.

Mitmproxy was initialized to generate the root CA certificate, which was then hosted on a local HTTP server for test devices to access and install (see Figure B3). For the iPhone XS, a semi-untethered jailbreak was performed with Dopamine (From TrollStore) to enable the installation of the Frida server. The device was connected to the laptop to verify Frida was able to communicate with the device, using the frida-ps command.

The A8 was connected to the laptop after enabling USB debugging. The HTTP server was brought back up to allow Android devices to install certificates (See Figure B4) Connection was verified and the device architecture was confirmed - before the matching frida-server version was downloaded and pushed to the device via ADB. Finally, the frida-server was initialized (see Figure B5).

Phase 3: Network Traffic Capture and Communication

Before capturing the communication session, the SSL unpinning script was initially run on the Galaxy A8 in capture mode, while it was connected to a standard network (see Figure C1).

This enabled the capturing of certificates seen during regular operation - a necessity before running the script in unpin mode. Different features in Snapchat were tested to generate the traffic for all modes of communication (see Table 5).

Table 5. Snapchat communication features tested in the controlled capture.

Category	Feature Tested
Messaging	Sent photo, and video snaps, received and opened them.
Media	Posted stories, viewed others' stories, saved snap to memories, loaded memories
Chat	Sent text messages, photo, and voice note in chat
Discovery/Explore	Browsed Discover page, watched Spotlight videos, loaded SnapMap
Account Operations	Login, logout, load friends list, search for users

After the initial capture session - all four devices were connected to the software AP. With the formal capture underway an mitmproxy session and tcpdump was started on both the AP interface and

internet-facing interface to capture packets (see Figure C2 and C3). The snapchat ssl pinning bypass was re-initialized, now in mode 1, to actively unpin requests. All Snapchat features previously tested were performed on the four test devices within the controlled communication setup. Each device communicated with each other, sending text messages, photo and video snaps, stories, and voice notes.

The mitmproxy session was run without the -w flag, so flows were not automatically written to the disk during live operation. Mitmproxy did receive the redirected Snapchat traffic from the PREROUTING rules, as evidenced by failed Snapchat domain connection attempts in the mitmproxy console during the capture session. The failures suggested Snapchat's certificate pinning mechanism rejected mitmproxys certificate - indicating the Frida SSL unpinning scripts failure to bypass pinning. Any detailed data on errors was not preserved due to the missing -w flag, and the manual saved file contained only the successful flows. All of which were HTTP requests that did not require any certificate pinning for completion. However, the tcpdump PCAPs on both interfaces were successful and contained all the network traffic data.

Phase 4: Device Acquisition

After the network capture session was completed, forensic images were acquired from all four devices. A full file system extraction was obtained from the Samsung Galaxy A8 using adb commands and Magnet AXIOM for a content-level extraction. A root shell was opened through adb shell with su (see Figure D1), the /data partition was then mounted, and a tar archive generated using command "tar -cvpf /sdcard/data_fs.tar/data (see Figure D2). Once the archive completed, it was pulled onto the forensic workstation using adb pull. The file integrity was verified by comparing hash values afterwards (see Figure D3).

The Samsung Galaxy S8 also went through two acquisition methods. First, a logical extraction was attempted through the Magnet AXIOM Process, which rendered very limited data to the non-rooted file structure. After realizing there were little to no Snapchat-related artifacts recovered with a logical extraction, the project's RCMP mentor assisted with a full file system extraction. GrayKey and Cellebrite UFED Premium were used to generate an Lx01 image which was parsed in FTK Imager to extract the data. The extracted data was then processed with Magnet AXIOM.

For the iPhone 13 and XS, encrypted iTunes backups were created. The devices were connected to a Macbook to perform the encrypted backup process using an encryption password (Figure D4). After backup completion, the files were transferred onto the primary forensic workstation where they were fed into Magnet AXIOM with the encryption password. Cellebrite Inseyets was also used as a secondary analysis tool. A separate case was created along with the same encryption password provided with the processing settings. (Figure D5)

Phase 5: Analysis and Cross-Referencing

Analysis was performed on all three acquisition areas. Network traffic was analyzed in Wireshark through the exported PCAPS, primarily looking at DNS queries, TLS handshakes, QUIC payloads, and communication timings. Android device data was analyzed using Magnet AXIOM Examine, along with database examination and confirmation in DB Browser for SQLite. Analysis of iOS devices were also done in Magnet AXIOM Examine and Cellebrite Inseyets, validating any artifacts found in both tools.

All options considered and attempted were documented properly throughout the whole analysis phase. When the iOS iTunes backups were extracted for the first time, it was realized that the backup improperly extracted all the data from the devices. This was noticed because the encryption password was not being prompted when adding the evidence in Magnet AXIOM Process. This therefore required re-extraction, which was successful on the second attempt. When the Galaxy S8 device failed to provide sufficient Snapchat-related evidence with a logical extraction, a full file system extraction was obtained. This was done in GrayKey and Cellebrite UFED Premium. All of these adaptations were documented in order to provide transparency throughout the investigation process.

Results

This section will present all the findings from each of the acquisition areas investigated. Results will be organized by their respective investigation method: network traffic analysis, Android device forensics, and iOS device forensics. All findings are presented without interpretation; analysis and discussion of the results will be found in the discussion section.

Network Traffic Analysis Results

Two PCAP files were extracted from the network captures in the controlled communication session: ap0capture and wlp61s-capture. The ap0capture provided the strongest evidence, and was analyzed in Wireshark. Three devices were identified on the network during the capture. Snapchat server IP addresses were identified, including 35.190.43.134 and 35.244.195.33 (Snapchat GCP servers), and 3.163.245.4 (Snapchat AWS media server). See Table 6 for device information found in the packet capture, including packet counts and data exchanges with Snapchat servers.

Table 6. Wireshark packet capture device information, packet counts, and exchanges

Device	IP Address	QUIC Packets Sent	QUIC Packets Received	Snapchat Servers Contacted	First QUIC Frame
iPhone XS (Sam)	192.168.100.44	5,142	12,329	GCP, AWS	12296

iPhone 13 (Dean)	192.168.100.91	3,849	8,938	GCP, AWS	40539
Samsung Galaxy A8 (Ryoma)	192.168.100.90	5,481	9,829	GCP, AWS	1902

Evidence of Data Transmission

Frame 1902 was the first encrypted QUIC packet transmitted during the capture. The Galaxy A8 (192.168.100.90) sent a packet of 1,292 bytes to Snapchat's server (35.190.43.134) using QUIC. This was the first packet sent from any device during the captured session, the Galaxy A8 (Ryoma) initiated Snapchat usage at this timestamp (see Figure 2).

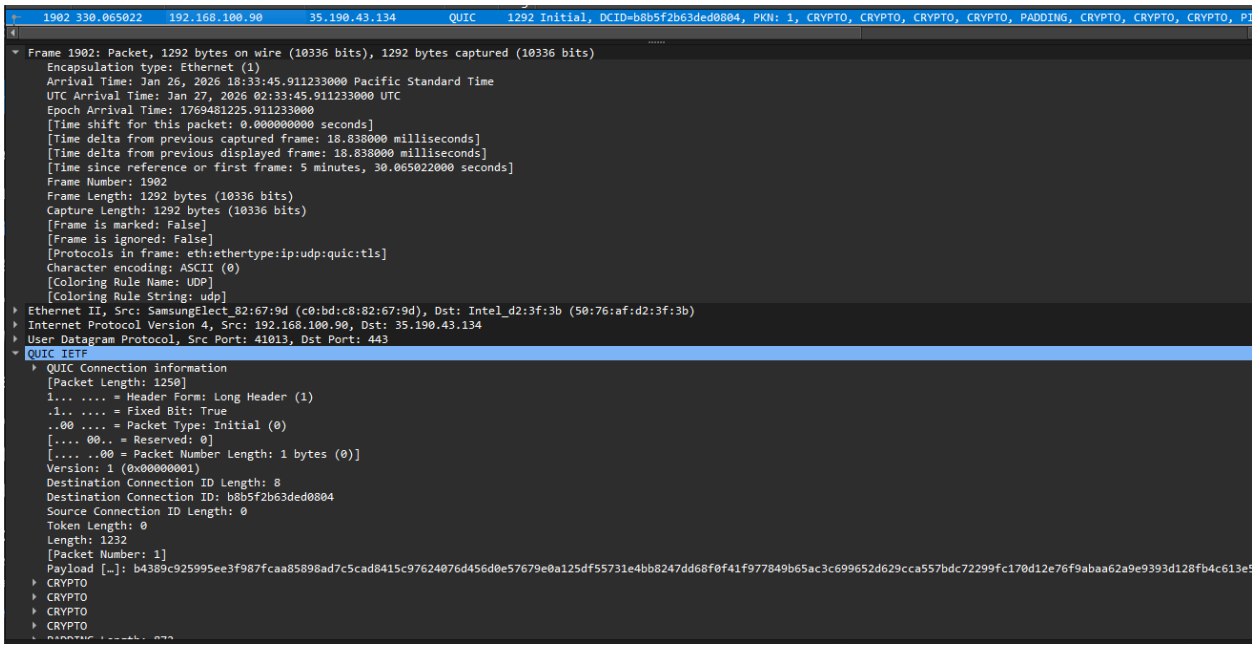


Figure 2. Wireshark packet capture of frame 1902 showing Galaxy A8's first encrypted QUIC payload to Snapchat server 35.190.43.134.

Next, frame 2032 contained data sent from Ryoma to IP address 35.244.195.33, which was identified as another Snapchat server (see Figure E1). This frame contained a QUIC protected payload as well. This provided evidence of user-generated data transmitted.

For the iPhone XS, the first QUIC frame was 12296, and it contacted Snapchat server 35.244.195.33. In frame 12297, it communicated with Snapchat server 35.190.43.194 (see Figure E2). On the iPhone 13, the first QUIC frame was 42954. Dean's device then contacted the Snapchat server 35.190.43.134 (see Figure E3).

Evidence of Data Reception

Frame 2338 had a larger size of 1,494 bytes that the A8 received from IP address 3.163.245.4 (identified as Snapchat AWS media server). This payload was also protected by QUIC (see Figure 3).

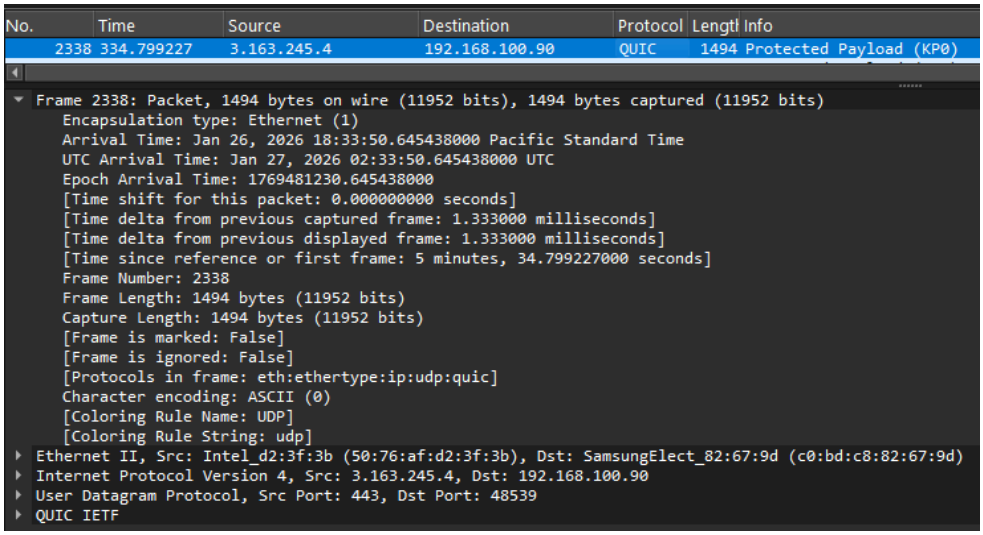


Figure 3. Wireshark capture frame 2338 containing 1,494 bytes of encrypted data from Snapchat's server on the Galaxy A8.

For the iPhone XS, frames 25912 and 25918 represented first contact with the AWS media server. This was where the initial media transfer to Sam began (see Figure E4). For the iPhone 13, frames 45185 and 45186 were the first contact with the AWS server. Frame 45186 is the first received packet (see Figure E5).

Evidence of Cross-Communication

Frames 1902 and 1991 represented instances where the Galaxy A8 and Snapchat server communicated (see Figure 4 and 5). This confirmed active communication rather than passive content loading. A communication pattern that was continuous was also identified from frames 2032 and 2042 (see Figures E1 to E3). Frame 2032 showed the A8 uploading data to Snapchat's server. Frame 2040 had the Snapchat server replying with a QUIC payload. In frame 2042, the A8 acknowledged the transmission. These frames confirmed an interactive communication.

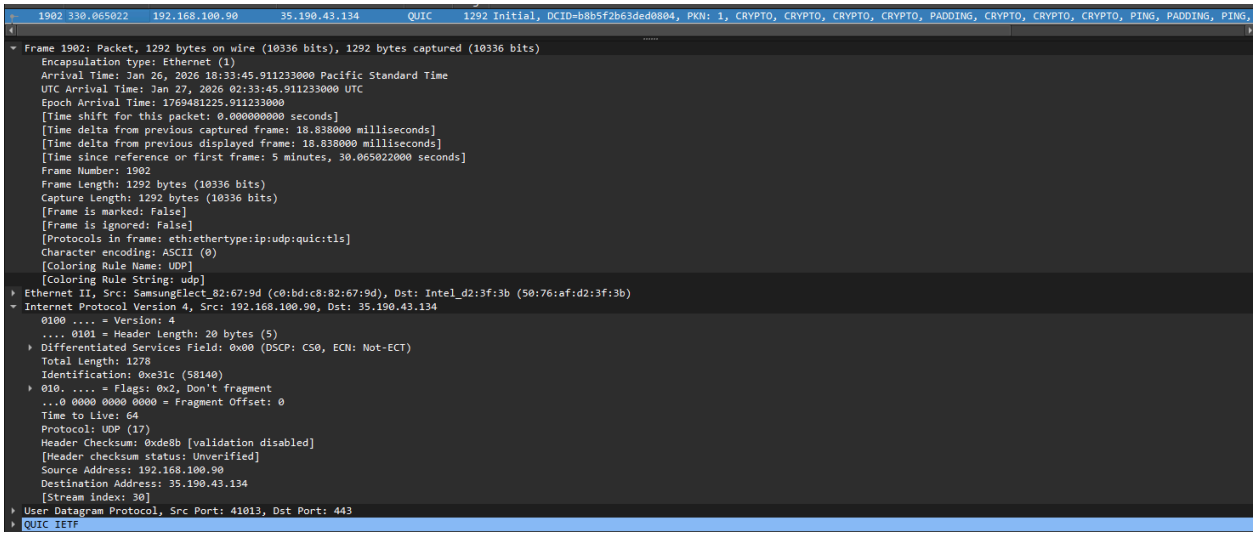


Figure 4. Frame 1902 containing communication data between Galaxy A8 and Snapchat server.

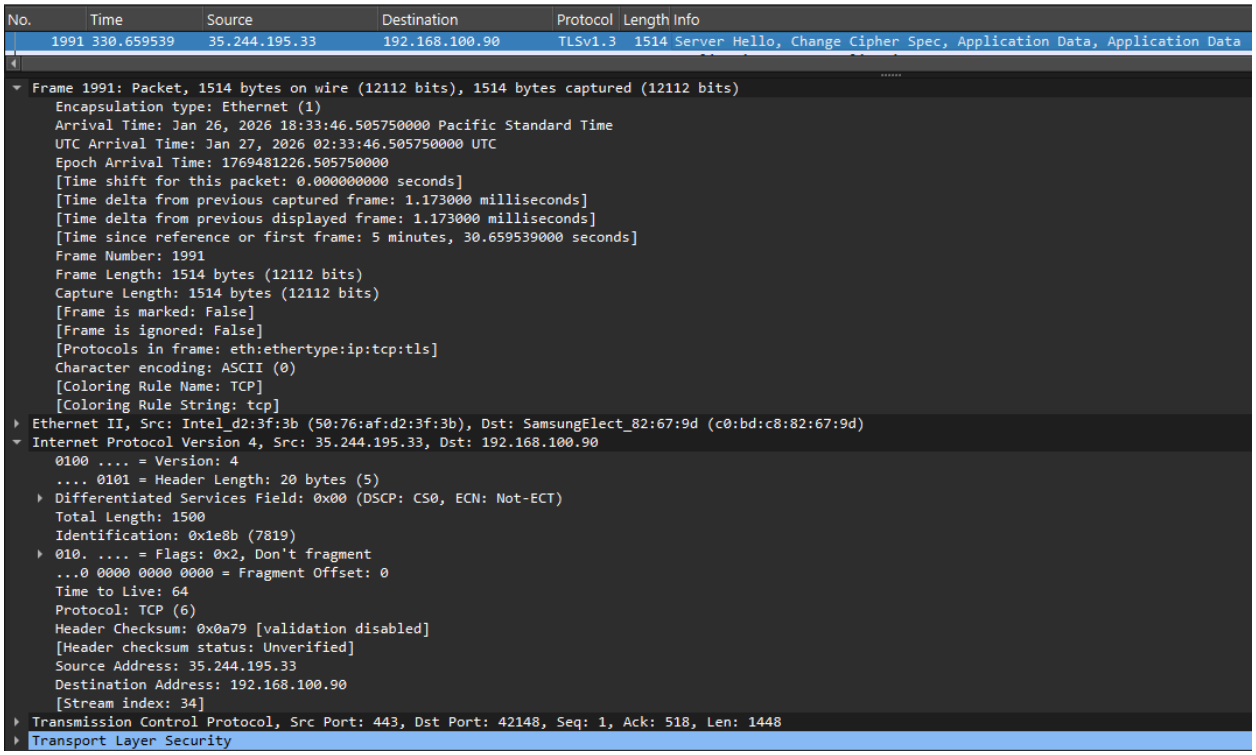


Figure 5. Frame 1991 showing communication from Snapchat server to Galaxy A8.

Frames 12296, 12339, and 12349 all represent cross-communication on the iPhone XS with Snapchat's servers. Having the same structure as the A8, the iPhone has an upload, server response, and acknowledgement step (see Figure E8).

On the iPhone 13, there was also cross-communication that occurred at frames 42954, 42963, and 42968. This process was identical to the A8 and XS's exchanges, confirming active communication with Snapchat (see Figure E9).

Android Device Forensic Results

Samsung Galaxy A8 (Rooted, Full File System Extraction)

The rooted Samsung Galaxy A8 on Android 9 provided the best results next to the Galaxy S8 in terms of Snapchat-related artifacts. The full file system extraction was analyzed using Magnet AXIOM Examine. The following categories of artifacts were recovered. Table 7 summarizes what artifacts were recovered from the Snapchat communication features.

Table 7. Recovery status of Snapchat features tested on Samsung Galaxy A8

Feature	Artifacts Recovered	Location/Path
Photo snap	Snapchat artifact and cached image	\native_content_manager\ and file-manager\
Video snap	Snapchat artifact and cached MP4 file	\native_content_manager\
Receive and opened snaps	Cached media	\file_manager cache
Posted story	Story media cache	\file_manager\post_story_snap
View stories	Story cache	\file_manager\story_snap
Saved memories	Memory cache media	\file_manager\memories_media
Chat texts	Full content	arroyo.db
Chat photo	Cached images	native_content_manager\ and file-manager\
Voice note	Cached audio files	native_content_manager

See Figure 6 below for a photo snap cache that had been recovered from the A8. Screenshots for the other above recovered artifacts are recorded in Appendix F.

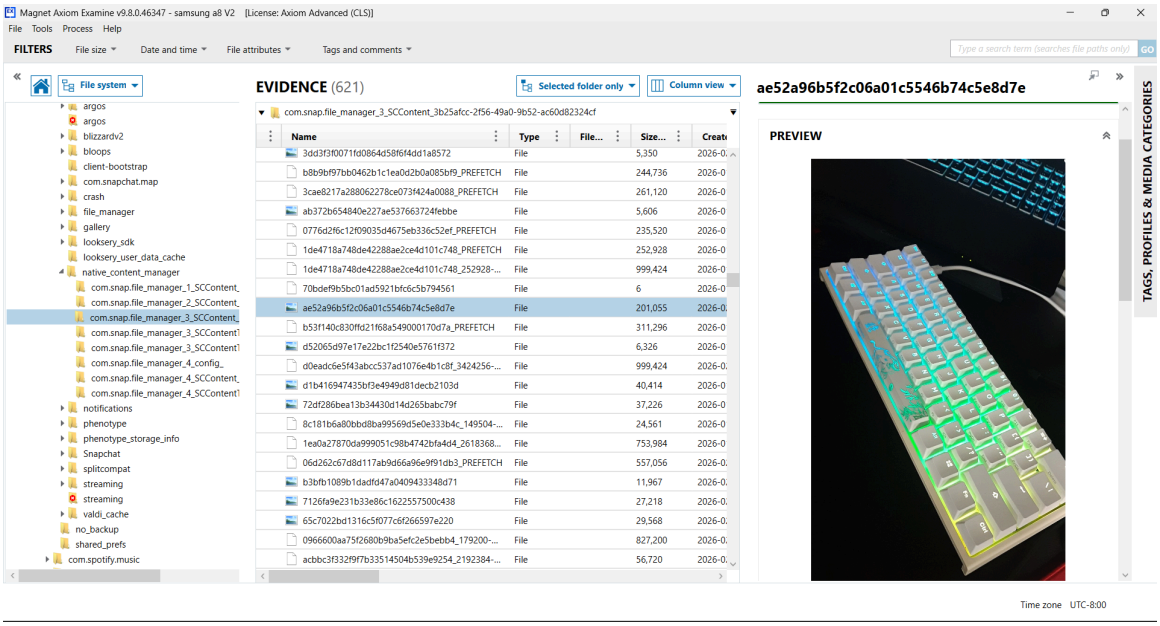


Figure 6. Photo snap cache recovered in Magnet AXIOM Examine.

Samsung Galaxy S8 (Non-Rooted, Logical Extraction)

When the Samsung Galaxy S8 was analyzed in a non-rooted state, it yielded no Snapchat-related evidence to prove anything was sent or received. The data was well protected regardless of having the password to the device and keeping it unlocked. All artifacts including databases like arroyo.db and main.db, and media such as videos, photos, or texts could not be recovered. Regardless of the lack of Snapchat-related evidence, a thorough analysis of any mention of Snapchat was performed. Some keyword searches included “Snapchat”, “picaboo”, and various media types (mp4, jpeg, png, etc.). What was found were documents containing “com.snapchat.android” in various different documents related to the phone’s operations. These included files containing information about usage statistics, accounts, activity, application operations, app widgets, battery information, and several others. Additionally, the User ID 10203 consistently showed up in some of these documents, suggesting it correlates to John Titor. However, this UID was never confirmed as John’s ID across the Android devices.

Samsung Galaxy S8 (Full File System Extraction via Graykey/Cellebrite)

With the assistance of the project mentor, a full file system image was obtained using GrayKey and Cellebrite UFED Premium. This image provided the second most amount of Snapchat-related evidence. The resulting Lx01 image from Cellebrite UFED Premium was extracted through FTK 7.3 and processed in Magnet AXIOM Examine.

Table 8 will summarize the Snapchat-related artifacts that were successfully recovered from the full file system extraction in Magnet AXIOM Examine. Following the table, screenshots of the evidence recovered will be found below the table. This covers all artifacts found.

Table 8. Snapchat artifact recovery results from non-rooted Samsung galaxy S8 using Magnet AXIOM Examine

Feature	Artifacts Recovered	Location/Path
Photo snap	Photo snap content and media cache	Snapchat Chat Messages and \file_manager\media\
Video snap	Video snap content and video cache	Snapchat Chat Messages and native_content_manager\
Receive\open snaps	Photo snap and media cache	file_manager\media\
Posted story	Story category, cached snapshots, thumbnail cache, and media cache	\native_content_manager\, \file_manager\media_package_thumb\, and posted_story_snap
Saved memories	Database confirmation and media cache	memories.db and memories_media
Chat text messages	Full message content	Snapchat Chat Messages
Photo sent in chat	Full snap content and media cache	Snapchat Chat Messages, native_content_manager, and DCIM\Camera
Voice note	Full voice note content and media cache	Snapchat Chat Messages and native_content_manager

See Figure 7 below for a photo snap cache file recovered from the media folder. Screenshots for the other above artifacts recovered can be found in Appendix G.

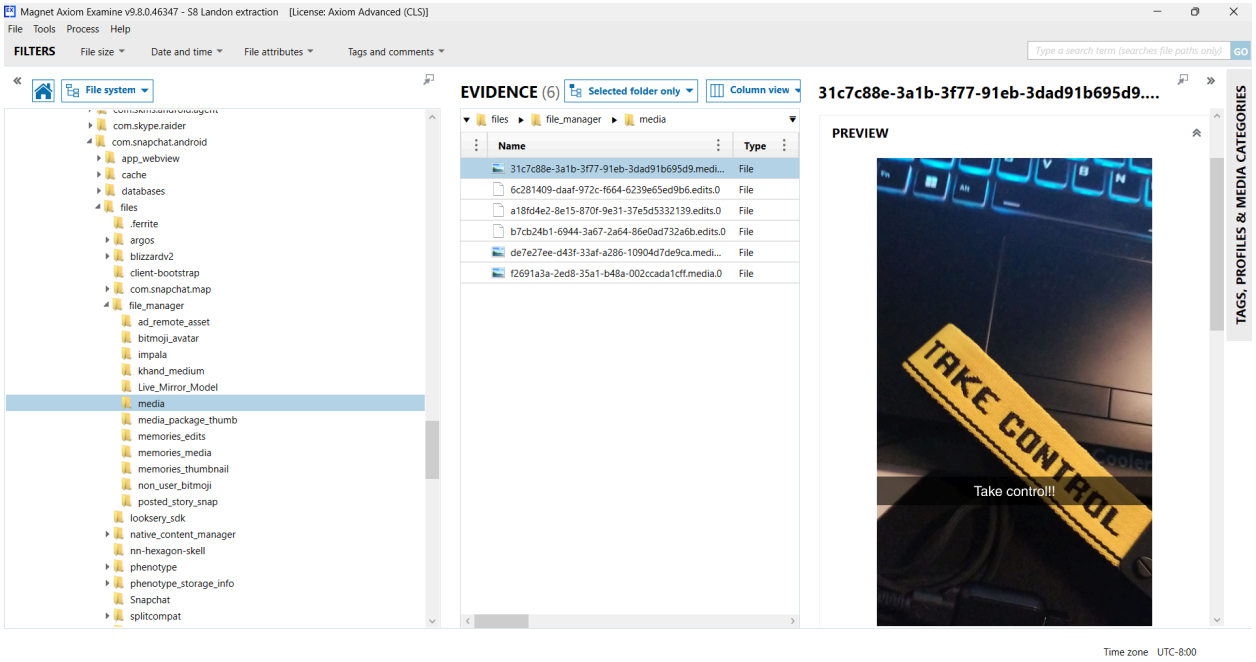


Figure 7. Photo snap media cache recovered on Galaxy S8 in media.

All Snapchat cache was commonly showing up in the following directories: /data/data/com.snapchat.android/files/native_content_manager, /data/data/com.snapchat.android/files/file_manager. These directories were consistent between both the Samsung Galaxy A8 and S8 extractions.

iOS Device Forensic Results

Device Identification and Ownership

Dean and Sam Winchester's devices were fully identified through artifacts in Axiom (see Figure 8 and H1). Parsed device information confirmed the following details (see Table 9). Snapchat application metadata was recovered for both devices as well (see Figure H2).

Table 9. Dean and Sam Winchester's device details

Artifact	Details (Dean W)	Details (Sam W)
UDID	00008110-000A48212228401E	00008020-0010549C3CC3002E
IMEI	355616300240508	353147101448472
Serial Number	WK92XXMC73	G0NZX3W6KPFP
Model/Model ID	iPhone 13 (iPhone14,5)	Model iPhone XS (iPhone11,2)
Device Name	dean	sam
Apple ID	deanw1967@proton.me	samw1967@proton.me
Backup Date	2026-02-19 6:16:26 AM UTC	2026-02-19 5:53:25 AM UTC

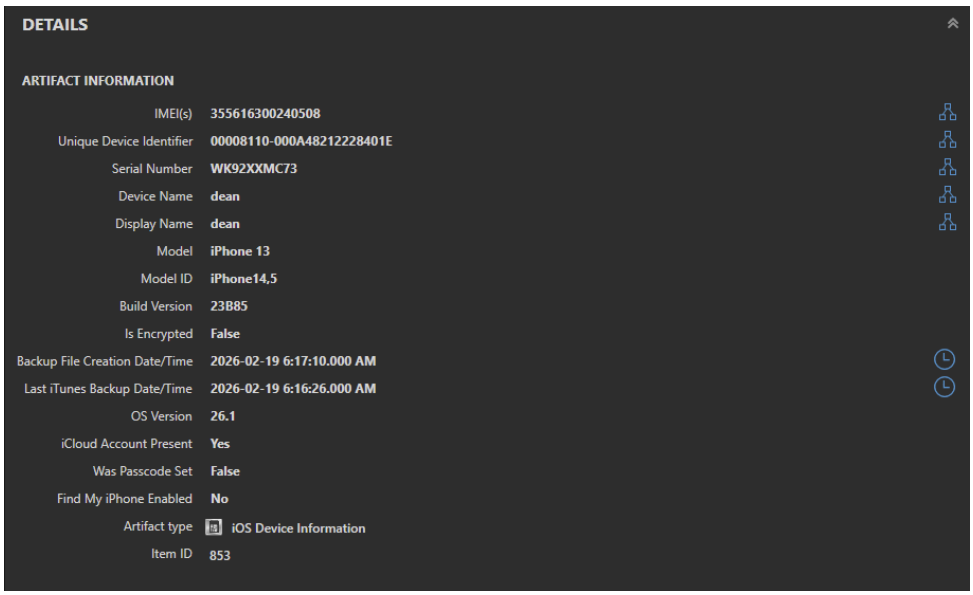


Figure 8. Axiom artifact showing Dean Winchester's device identification details

Dean's iPhone 13 had Snapchat version 13.79.1 installed at /private/var/mobile/Containers/Data/Application/7B5CD489-B3DF-438D-A1AB-711F90C829FD\.

Sam's iPhone XS had Snapchat version 13.75.0 at /private/var/mobile/Containers/Data/Application/B325439C-0A6D-4745-BECF-F1E45AFFF343\.

Snapchat Keychain Artifacts

Three keychain properties were found for each Snapchat account. These properties were: SCOneTapLoginKeychainKey, fideliusTransferableDeviceGraph, and fideliusTransferableIdentityBackup (see Figure 9). Inference was required based on naming conventions and supporting research, as no publicly available documentation existed for these exact keychain properties during analysis.

Artifact	Key detail	Supporting detail	Additional detail	Date and time	Ite...
Encryption & Credentials Apple Keychain Generic Passwords	Keychain Property fideliusTransferableDeviceGraph	Value 5453414603000400 05...	Service Name com.toyopagroup.picaboo	Created Date/Time 2026-01-22 8:20:10.558 AM	51004
Encryption & Credentials Apple Keychain Generic Passwords	Keychain Property fideliusTransferableIdentityBackup	Value 5453414603000400 03...	Service Name com.toyopagroup.picaboo	Created Date/Time 2026-01-22 8:20:29.584 AM	51005
Encryption & Credentials Apple Keychain Generic Passwords	Keychain Property SCOneTapLoginKeychainKey	Value {"$version":100000,"$a...	Service Name com.toyopagroup.picaboo	Created Date/Time 2026-01-22 8:20:30.706 AM	51006

Figure 9. Axiom keychain artifacts for Sam Winchester's Snapchat account on the iPhone XS (Items 51004-51006)

Based on the earliest fideliusTransferableDeviceGraph artifact timestamp, Sam Winchester activated his Snapchat account on January 22, 2026 at 8:20:10 AM UTC. Dean Winchester activated his Snapchat account five days later, on January 27, 2026 at 2:00:39 AM UTC (See Figure H3). The SCOneTapLogin session tokens for Sam and Dean were both created a minute later.

SCOneTapLoginKeychainKey

Apple's Authentication Services framework provides credential storage and one-tap authentication for iOS applications, as documented in Apple's developer resources and presented at WWDC 2020 (Apple, 2020; Apple Developer Documentation, n.d). Support documentation from Snapchat affirms that they offer this one-tap method, and state users who choose to utilize this feature will have their account appear on the login screen. This allows for simple one-tap authentication on subsequent logins using the saved credentials (Snapchat Support, n.d.). The "SC" prefix in the artifact appears to directly correlate to the Snapchat application, as supported by the attached Service Name "com.toyopagroup.picaboo". The documented one-tap authentication feature directly aligns with the "OneTapLogin" artifact. While this exact keychain property does not appear in any publicly available documentation, but the naming convention and its direct relation to the Authentication Services framework supports that this entry stores Snapchat's login credentials. The timestamps also correspond to known session authentication events on each device.

Fidelius-Prefixed Properties

The fideliusTransferableDeviceGraph and fideliusTransferableIdentityBackup artifacts share the "fidelius" prefix, which was confirmed to be an internal Snapchat service through a GitHub Go Library

created to interact with Snapchat's Web API. The service, "snapchat.fidelius.FideliusIdentityService", exposes two endpoints - GetFriendKeys and InitializeWebKey (see Figure 10). InitializeWebKey was confirmed by the source code to generate a public/private cryptographic key pair for the device (Figure H4). Based on the presence of these cryptographic key generation and retrieval endpoints, the Fidelius service is inferenced to function as Snapchat's end-to-end encryption identity and key distribution system. Snapchat has not officially published details on encryption protocols or schemes, making further verification of their internal architecture difficult (Bhuse, 2023). However, Snap Inc. security engineers confirmed that E2EE for Snaps was introduced in January, 2019 (Sankuratripati et al., 2019).

The "TransferableIdentityBackup" naming convention is consistent with Apple's practice of storing cryptographic identities within the iOS Keychain (Apple Developer Documentation, n.d). The "Transferable" prefix aligns with research on Apple's iCloud Keychain framework, which syncs keychain items to new devices through a device trust circle - which allows keychain information to be migrated to a new device (Apple, n.d). The "TransferableDeviceGraph" property is assessed to represent a map of authorized devices, based on the Fidelius key generation architecture and Snapchat's presentation showing per-device public key registration as part of their E2EE solution. This appears consistent with other established E2EE multi-device architectures, such as WhatsApp's engineering analysis which uses a similar design approach (Meta Engineering, 2021).

```
31    var FIDELIUS_SERVICE_URL = WEB_BASE_URL+"/snapchat.fidelius.FideliusIdentityService"
32    var INITIALIZE_WEB_KEY = FIDELIUS_SERVICE_URL+"/InitializeWebKey"
33    var GET_FRIEND_KEYS = FIDELIUS_SERVICE_URL+"/GetFriendKeys"
```

(Oxzer, 2023) **Figure 10.** Code snippet from Github Repo with the GetFriendKeys and InitializeWebKey endpoints

Interaction C Communication Records

Under application usage, the InteractionC database included communication events on both iOS devices - logging interactions between contacts. It did not, however, preserve message content within the backup. The records included timestamps, contact identifiers, and interaction types. Artifacts between Ryoma Echizen and Sam show the creation of the "Ryoma Echizen" contact in InteractionC Contacts and the first occurrence of communication in InteractionC Interactions almost simultaneously, within .001 of a second (see Figure 11).

Application Usage InteractionC Contacts	Identifier 3b25afcc-2f56-49a0-9b52-ac60d82324cf	Display Name Ryoma Echizen	Incoming Interaction Count 0	Created Date/Time 2026-01-27 1:10:34.595 AM	57196
Application Usage InteractionC Interactions	Bundle ID com.toyopagroup.picaboo			Created Date/Time 2026-01-27 1:10:34.594 AM	57206

Figure 11. Items 57196 and 57206, showing initial contact between Sam and Ryoma

Almost a month later, Snapchat's included "My AI" and Sam speak for the first time (see Figure 12).

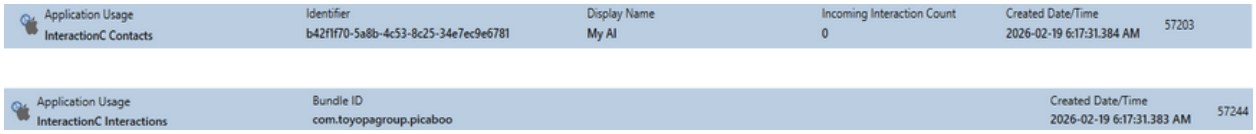


Figure 12. Items 57203 and 57244, showing initial contact between Sam and My AI

On Dean's device, the densest cluster of InteractionC events (40 events) occurred between items 6986 and 7026. This represented a 35 minute conversation between 3 AM and 3:35 AM on February 13th (see Figure H5/6/7). While interactions show a conversation occurred, no message content was preserved alongside this information. Dean and Sam's final conversation occurred on February 19th, with 11 recorded events in a 3-minute window at 6:10 AM (see Figure H8).

While not relevant to evidence in this case due to their non-personalized nature, InteractionC events also recorded notifications sent by the Snapchat application. This included video and lens suggestions, as well as discover and following page post notifications (see Figure H9).

Media and Location Artifacts

Photos saved in Snapchat were recovered from both iOS devices. Cellebrite Inseyets found these photos in the camera roll. These showed a capture origin of "Saved Copy" - with the images themselves being inferenced to be Snapchat-originated based on the text caption style (see Figure 13). These images were saved to both Snapchat memories and the device's camera roll.

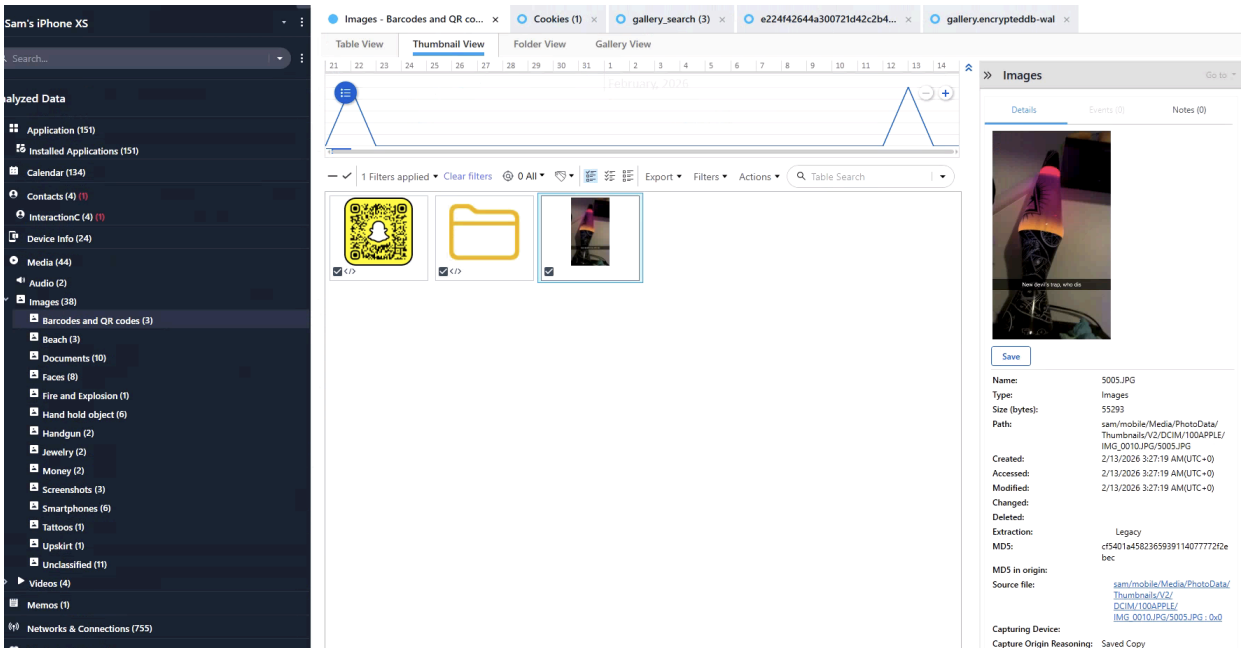


Figure 13. Recovered photo in Cellebrite Inseyets as a "Saved Copy" on iPhone XS.

Magnet Examine directly identified images as having a Snapchat-origin, attaching Snapchat's application name and bundle ID to the image (see Figure 14). This is the exact same image seen above in Cellebrite Inseyets.

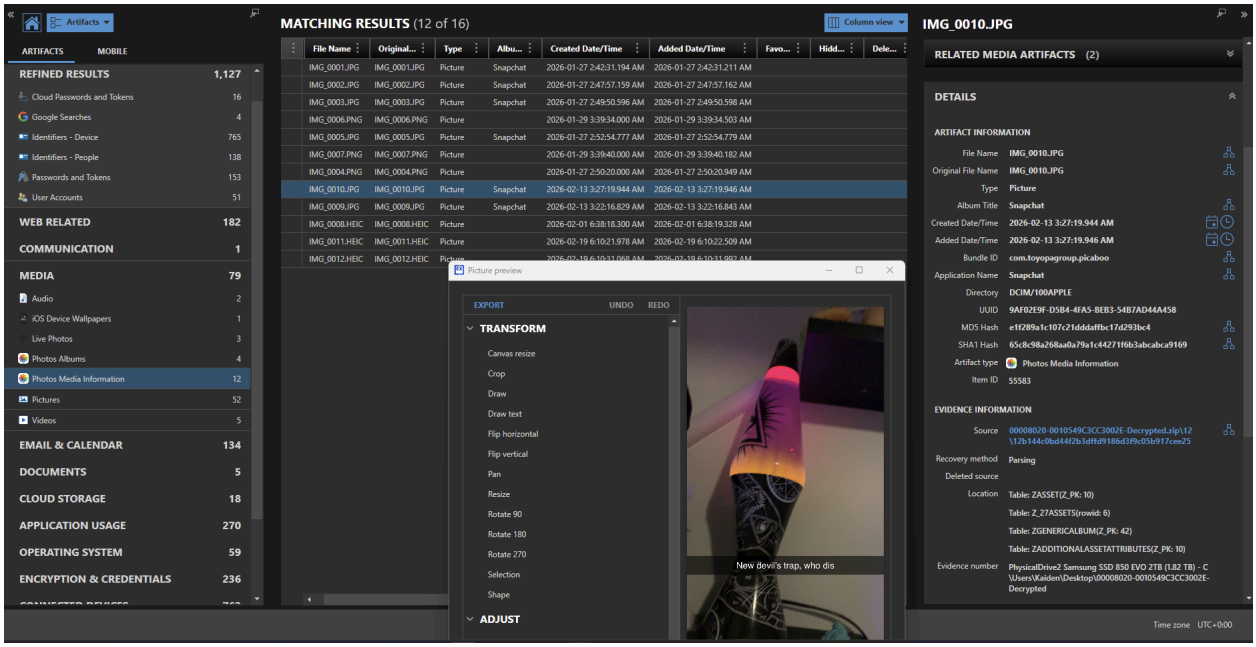


Figure 14. Recovered photo in Magnet Examine as Snapchat-origin item, with Application Name and Bundle ID

Cellebrite Inseyets location carving feature provided extensive information on device location data, with a dedicated section to aggregate all hits into a single consolidated view. Fifty location hits from Sam's device matched one geographic location in Maple Ridge, British Columbia (see Figure 15)

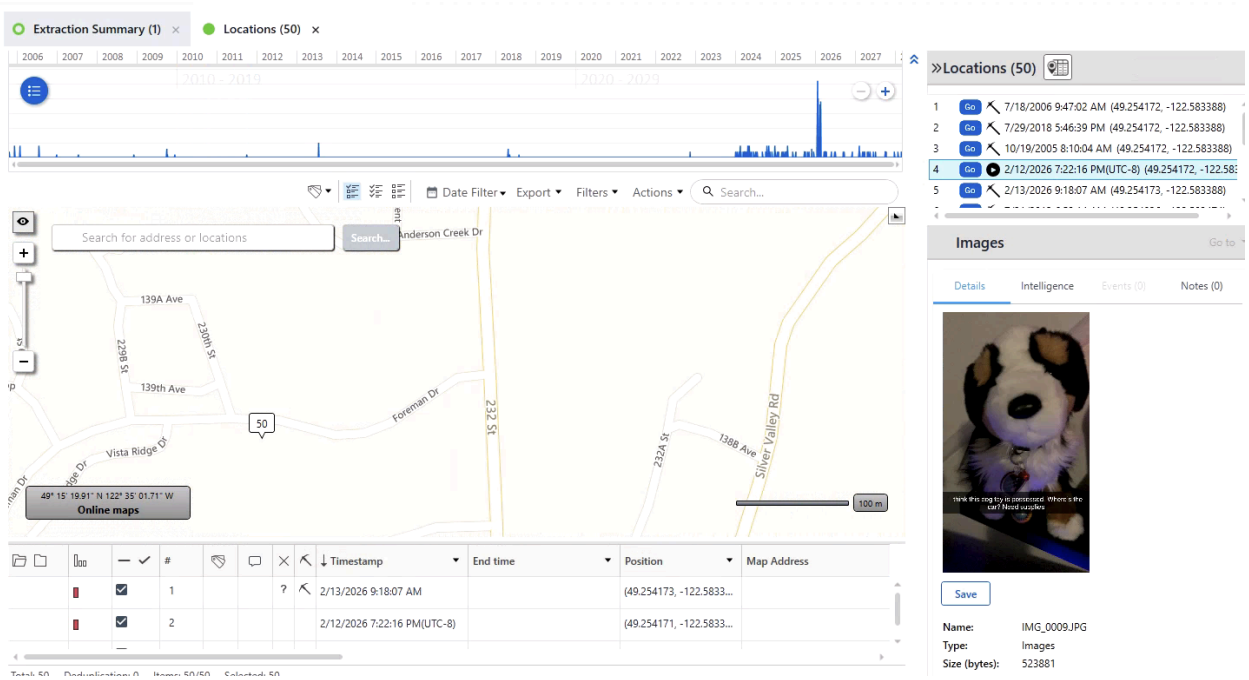


Figure 15. Cellebrite Inseyets location carving results showing consolidated location data from Sam's iPhone XS

A screenshot taken from Dean's iPhone 13 was also identified. The screenshot was taken soon after the last recorded message in the conversation between Dean and Sam. EXIF metadata embedded in the screenshot revealed that it was taken in Maple Ridge, British Columbia, with precise GPS coordinates (see Figure H10).

Cellebrite Inseyets presented artifacts in a different structure than AXIOM. Contacts from InteractionC were identified in a dedicated section rather than being grouped with InteractionC events (Figure H11). However, display names were absent from Inseyets presentation of InteractionC events, requiring manual matching of events to known accounts (Figure H12).

Application and System Metadata

Magnet Axiom's connection view showed connections between the iPhone Xs' info.plist configuration file and Snapchat-related artifacts (see Figure 16). The mapping connected evidence of Snapchat usage alongside jailbreaking software artifacts (Dopamine, Trollstore) and device identifiers. This provided additional context about the device's state at the time of the backup, confirming semi-untethered jailbreak software was present during acquisition.

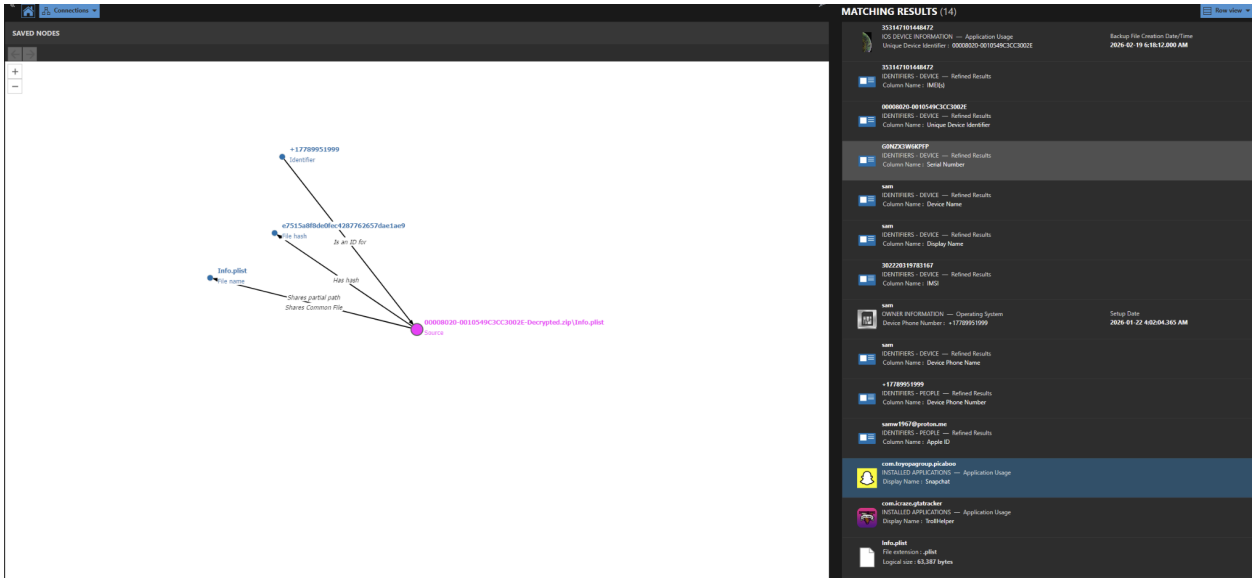


Figure 16. Info.plist mapping of Sam's iPhone XS - showing connections between Snapchat usage, jailbreak software, and device identifiers

Encrypted Content Limitations

A gallery_encrypted directory was identified under the Snapchat application's data grouping on both iOS devices (see Figure 17). The content inside could not be accessed or decrypted by only using the encrypted backups - something that would have been possible with a Magnet GrayKey acquisition. This directory appears to directly correspond to Snapchat's My Eyes Only feature.

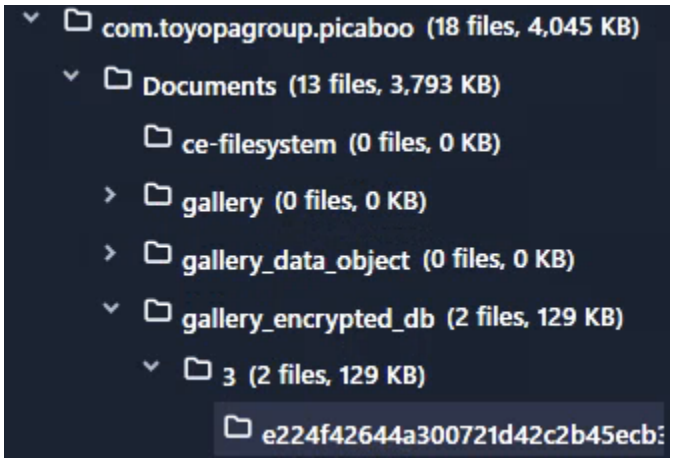


Figure 17. Cellebrite Inseyets showing the gallery_encrypted directory under Snapchat application data, with inaccessible content

Cross-Platform Comparison

Table 10. Cross-platform Snapchat artifact recovery comparison

Artifact Type	A8 (Root FFS)	S8 (Logical)	S8 (GrayKey FFS)	iOS (iTunes Backup)
Chat Messages	Full content	None	Full content	Metadata only
Photo Snaps	Cached media	None	Cached media	Saved copies only
Video Snaps	Cached media	None	Cached media	Not recovered
Stories	Media + DB entry	None	Thumbnail + cache	Not recovered
Voice Notes	Audio file	None	Audio file	Not recovered
Memories	Media file	None	Media file	Encrypted (MEO)
Contact Lists	Database entry	None	Database entry	InteractionC
Timestamps	Full precision	Indirect only	Full precision	Full precision
Location Data	Not applicable	Not applicable	Not applicable	EXIF + carving
Account Info	Database entry	Email in logs	Full profile	Keychain tokens
User IDs	In databases	None	Confirmed in DB	InteractionC IDs

Discussion

The research question that guided this study was: What data can be recovered in an investigation where Snapchat is the primary source of evidence? The discussion following interprets results from each different acquisition vector. It relates to findings in existing literature, and identifies study limitations and areas to be improved on for future research.

Network Traffic Analysis

Network traffic analysis results showed that there are benefits and limitations of network forensics for Snapchat. Data analyzed verified communication between the mobile devices and Snapchat servers. However, encryption methods like TLS1.3 and QUIC prevented recovery of the data contents, limiting what could be confirmed. This realization ended up being consistent with the literature review's assessment for Snapchat's use of encryption and network security, which resulted in great limitations of the potential to observe external user data (Snap Engineering, 2021; Luxemburk & Hynek, 2023).

Regardless of the encryption and network protection, the network capture analysis provided substantial evidence for forensic metadata. DNS queries to Snapchat domains like gcp.api.snapchat.com and aws.duplex.snapchat.com confirmed that the devices actively communicated with Snapchat's infrastructure. Sizes of packets and rapidly sent packets also provided indications of the content type

being transmitted such as content delivery. TLS handshake data with Server Name Indication fields naming Snapchat domains also provided proof. This metadata correlates with the forensic principle discussed by Heath et al. (2023), which is that temporary metadata still has valuable evidence even if content is not recoverable. The metadata found, has potential to confirm application usage and device-level findings in an investigation. However, after a successful forensic analysis of mobile device artifacts, it helps to confirm that the forensic tools alone can provide evidence of timelines.

Android Device Forensics

Results from the Android analysis revealed a huge difference in access levels between a rooted and non-rooted device. The rooted Galaxy A8 full file system extraction provided artifacts from all tested Snapchat features. This included full chat message content, cached photo and video snaps, and story, memory, and voice note cache. All findings were consistent with prior research from Alyahya and Kausar (2017) and Aji et al. (2017). These documented Snapchat artifacts are in the Snapchat application folder on Android devices. This study was also able to confirm persistence of artifacts in databases like arroyo.db and main.db, compared to an older tcspahn.db mentioned in the literature review. From the full file system extraction on the Galaxy S8, artifacts recovered were consistent with those found on the A8.

A majority of cached media locations that were found on the Galaxy S8 and A8 were consistent. This included /data/data/com.snapchat.android/files/native_content_manager, and /data/data/com.snapchat.android/files/file_manager. The consistency of these findings across two devices on different Android versions suggest the paths remain as reliable locations for Snapchat evidence to reside. Although, future application updates and newer Android versions are prone to have different results, considering relatively older versions were only tested.

iOS Device Forensics

The iOS analysis revealed much more limited information than the Android devices, but what was recoverable should still be seen as forensically important. The encrypted iTunes backups provided metadata for conversations within the InteractionC databases, keychain authentication artifacts, media with EXIF data, and information about the installed applications. Direct message content was unrecoverable from the backups. These results were consistent with findings from Bates and Karabiyik (n.d) as they documented similar limitations regarding iOS forensics - and with the broad observation from Azhar et al. (2020) that iOS devices present greater challenges due to sandboxing and file system encryption.

For timeline reconstruction, the InteractionC database proved particularly useful. The database recorded timestamps, conversation participants, and occurrences of communication. The densest cluster of events took place on Dean’s iPhone 13 - where over 35 minutes, 40 InteractionC (communication events, items 6986-7026) occurred. The final recorded conversation session had 11 interactions in three minutes on the 19th of February. Both provided evidence of communication patterns without revealing the literal

message content. The InteractionC database's behaviour of recording contact creation and interaction events provides a mechanism for establishing first contact between users on Snapchat, which may be valuable for establishing a timeline. This type of metadata could prove valuable for establishing behaviour patterns or corroborating other evidence sources - Habib et al. (2019) noted even this metadata can provide important context for forensic investigations.

The InteractionC database also included references to data that - in another case where the Snapchat application is used on a more consistent basis for media - may contain evidence that directly relates to what the suspect is interested in or subscribed to. Several events showed Sam Winchester communicating with Snapchat's personal AI chatbot, which retains shared interests and facts. Manual deletion is required to remove the information Snapchat stores from conversations. Interactions may very well provide significant additional data regarding Sam Winchester's habits, thoughts, and patterns if investigators have access to the account to download the data.

The keychain artifacts (SCOneTapLoginKeychainKey, fideliusTransferableDeviceGraph, fideliusTransferableIdentityBackup) were an area where original research was a necessity due to a lack of public documentation. The inferences from naming conventions and Apple developer documentation provided a reasonable platform for understanding the relevance of these artifacts. However, conclusions here are open for future discussion and revision until confirmed with additional research or disclosure from Snapchat.

The gallery_encrypted directory, assessed to correlate with Snapchat's My Eyes Only feature were a hard limitation encountered with the iTunes backup acquisition method. The encryption keys for this are stored within the iOS devices Secure Enclave, and they are inaccessible without a Magnet GrayKey extraction. This underscores the importance of using advanced acquisition tools to capture the full picture and obtain maximal evidence.

Magnet Axiom and Cellebrite Inseyets each had their own strengths for iOS analysis. Axiom presented InteractionC data more intuitively with display names and grouped interactions. Inseyets provided superior location data carving. With media identification, Magnet Examine directly attaches recovered images to the application name and bundle ID (com.toyopagroup.picaboo), whereas Cellebrite Inseyets classified the same content as a "saved copy" without connection to the application. This is significant as Magnet's attribution provides stronger evidence that the images originated from Snapchat. Both tools used in parallel enabled cross-validation of findings and a more complete picture overall.

Study Limitations

This study was conducted in a controlled lab environment with test accounts, full device access, and a predetermined list of actions taken. In the real world, forensic scenarios involve unknown usage patterns, legal constraints on acquisition, and potentially users who attempt to hide evidence they generate. The findings in this study represent a best-case scenario due to its controlled nature - organic investigations will experience more limitations.

With four devices - two Android and two iOS - findings cannot be generalized. Different devices, operating system versions, and application versions will vary with levels of recovery success. Snapchat updates its application on a frequent basis - it is very possible such artifact locations and encryption methods may change between versions. Study findings are current - as of the Snapchat versions tested (13.75.0 - 13.79.1).

The mitmproxy session log failure during the capture demonstrates an unplanned data loss. While the primary network data was preserved by the PCAP captures, the missing proxy error logs prevented determining why the SSL unpinning failed exactly. Proceeding replications should implement additional logging measures and verify all capture parameters function before commencing communication.

The inability to root the S8 device is an example of a real-world constraint that investigators might run into. Moving to obtain a full file system extraction through Cellebrite Premium may not be an option for all forensic use cases.

Recommendations for Future Research

Future research should explore Snapchat artifacts recovery on a wider range of operating systems and devices to assess the general applicability of these findings. Particular focus should be placed on newer Android versions (13 and above) that utilize more restrictive file system policies - and iOS full file system extractions that may yield content not discoverable in encrypted iTunes backups.

The keychain properties using a Fidelius prefix need further investigation. Snapchat does not publicly document these properties, so reverse engineering or direct consultation would be needed to confirm inferences made in this study. Future forensic methodologies may be better informed with an understanding of the role these artifacts play in Snapchat's architecture.

The network analysis methodology could be enhanced through the use of different interception tools that support QUIC as they improve. Mitmproxy's addition of HTTP/3 support suggests future versions may provide more effective inspection of QUIC-encrypted traffic, enhancing analysis of Snapchat network communications. Updated scripts for bypassing the certificate pinning Snapchat employs could significantly enhance traffic analysis with a direct capture of traffic.

Conclusion

To conclude, this study's objective was to determine what data is recoverable when Snapchat is the primary source of evidence. Research performed examined three different areas: network traffic interception, Android device forensics, and iOS device forensics. Results from the study support the following conclusions.

Snapchat's network security and encryption techniques are effective against forensic investigations. Even though using a controlled man-in-the-middle lab setup with mitmproxy and Frida certificate pinning bypasses, message content still remains unrecoverable. Snapchat's QUIC protocol along with application-level E2E encryption ensure this unrecoverability. However, network metadata such as DNS queries, server IP addresses, packet sizes, and timestamps were still viewable which provided useful forensic evidence.

Device access level is the main factor to determine if artifacts are recoverable on Android. Rooted devices were very clear in providing the most Snapchat content for artifact recovery. This included message content, cached media, database records, and metadata. On the other hand, non-rooted devices have no sufficient evidence for Snapchat. Full file system extractions and acquisitions performed with enterprise-grade tools like GrayKey, and Cellebrite Premium provided similar artifacts that a rooted device provides.

iOS encrypted backups provide metadata but not message content. Recoverable artifacts included: InteractionC records, keychain authentication artifacts, saved media with EXIF data, and application metadata. My Eyes Only encrypted content was not accessible without the correct decryption keys - a limitation of utilizing the encrypted backup feature for acquisition. iOS analysis also showed that InteractionC databases provided conversation timelines and records of first contact. EXIF metadata and location carving provided precise geographical data. The use of both tools (Magnet Axiom and Cellebrite Inseyets) provided a more complete artifact recovery than either on its own.

The cross-platform comparison showed that Android devices with full file system access delivered the most extensive Snapchat artifact recovery. iOS encrypted backups contained useful metadata and timeline information but did not include direct message content. This was consistent with the literature reviews assessment that Android devices are typically more forensically accessible for extraction (Azhar et al., 2020). Additionally, it supports Malley's (2021) conclusion that recovery outcomes varied significantly between device type and operating system design.

Ephemeral data remains in device storage. Despite Snapchat's design of disappearing content, significant traces of media and other artifacts were found across all devices when using the appropriate acquisition methods. Attributes like the operating system, device access level, and forensic tools all were factors to determine the recovery success. Findings from this study both contribute and confirm to the growing research on ephemeral messaging forensics, providing a practical guide for investigators handling Snapchat evidence in digital forensic examinations.

Citations

0xzer. (2023). *snapper/data/paths/paths.go* [Source code]. GitHub. https://github.com/0xzer/snapper/blob/main/data/paths/paths.go

Alyahya, T., & Kausar, F. (2017). Snapchat analysis to discover digital forensic artifacts on Android smartphone. *Procedia Computer Science, 109,* 1035–1040. https://doi.org/10.1016/j.procs.2017.05.421

Apple. (2020, June). *One-tap account security upgrades* (Session 10666) [Video]. Apple Developer. https://developer.apple.com/videos/play/wwdc2020/10666/

Apple. (n.d.-a). *Authentication Services.* Apple Developer Documentation. https://developer.apple.com/documentation/AuthenticationServices

Apple. (n.d.-b). *Secure keychain syncing.* Apple Platform Security. https://support.apple.com/guide/security/secure-keychain-syncing-sec0a319b35f/web

Apple. (n.d.-c). *Storing an identity in the keychain.* Apple Developer Documentation. https://developer.apple.com/documentation/security/storing-an-identity-in-the-keychain

Azhar, H., Cox, R., & Chamberlain, A. (2020). Forensic investigations of popular ephemeral messaging applications on Android and iOS platforms. *International Journal on Advances in Security, 13*(1 & 2), 41–53.

Barcaroli, L. (2023, August 11). *Inspect TLS encrypted traffic using mitmproxy and Wireshark.* Koyeb. https://www.koyeb.com/blog/inspect-tls-encrypted-traffic-using-mitmproxy-and-wireshark

Bates, S., & Karabiyik, U. (n.d.). *Snapchat forensics for iOS.* https://siennabates.com/pdfs/Snapchat_Forensics_for_iOS.pdf

Baxter, J. (2025). *A history timeline about Snapchat.* Historytimelines.co. https://historytimelines.co/timeline/snapchat

Bayer, J. B., Ellison, N., Schoenebeck, S. Y., & Falk, E. B. (in press). Sharing the small moments: Ephemeral social interaction on Snapchat. *Information, Communication & Society.*

Bhuse, V. (2023). Review of end-to-end encryption for social media. *International Conference on Cyber Warfare and Security.* https://papers.academic-conferences.org/index.php/iccws/article/download/1017/924

Certificates. (2025). *Mitmproxy.org.* https://docs.mitmproxy.org/stable/concepts/certificates/

Constine, J. (2019, January 29). *Facebook pays teens to install VPN that spies on them.* TechCrunch. https://techcrunch.com/2019/01/29/facebook-project-atlas/

Cyberly. (2025). *How does Frida assist in bypassing system-level protections in iOS and Android apps?* Cyberly.org. https://www.cyberly.org

Doris, A. (2022). *Best way to recover deleted Snapchat messages on iPhone.* Anyrecover.com.

Forensic Control. (2025, April 8). *ACPO guidelines & principles explained.* https://forensiccontrol.com/guides/acpo-guidelines-principles-explained/

Franceschi-Bicchierai, L. (2024, March 26). *Facebook snooped on users' Snapchat traffic in secret project.* TechCrunch. https://techcrunch.com/2024/03/26/facebook-secret-project-snooped-snapchat-user-traffic/

Habib, H., Shah, N., & Vaish, R. (2019). Impact of contextual factors on Snapchat public sharing. *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems.* https://doi.org/10.1145/3290605.3300256

Heath, H., MacDermott, A., & Akinbi, A. (2023). Forensic analysis of ephemeral messaging applications: Disappearing messages or evidential data? *Forensic Science International: Digital Investigation, 46,* 301585.

Hils, M. (2023, August 4). *Mitmproxy 10: First bits of HTTP/3!* Mitmproxy.org. https://www.mitmproxy.org/posts/releases/mitmproxy-10/

Jain, G., & Hils, M. (2024, October 4). *Mitmproxy 11: Full HTTP/3 support.* Mitmproxy.org. https://www.mitmproxy.org/posts/releases/mitmproxy-11/

Johnson, D. (2023, February). *How to recover deleted messages from a Snapchat account.* Alphr. https://www.alphr.com/recover-deleted-messages-snapchat/

Lopes, J. (2018, March 13). *Using Frida to bypass Snapchat's certificate pinning.* LRQA. https://www.lrqa.com/en/cyber-labs/using-frida-to-bypass-snapchats-certificate-pinning/

Luxemburk, J., & Hynek, K. (2023). Encrypted traffic classification: The QUIC case. *TMA Conference 2023.*

Mahalik, H. (2022, August 24). *How to use the Snapchat forensics features built into Physical Analyzer.* Cellebrite.

Malley, A. (2021). *A comparison analysis of saved Snapchat video files on Androids vs iPhones* [Unpublished paper]. University of Denver.

Meta Engineering. (2021, July 14). *How WhatsApp enables multi-device capability.* Engineering at Meta. https://engineering.fb.com/2021/07/14/security/whatsapp-multi-device/

Mukhlis Prasetyo Aji, Imam Riadi, & Lutfhi, A. (2017). The digital forensic analysis of Snapchat application using XML records. *Journal of Theoretical and Applied Information Technology, 95*(19).

Mutawa, N., Baggili, I., & Marrington, A. (2016). Forensic analysis of social networking applications on mobile devices. *Digital Investigation, 9,* S24–S33.

Nolan, A. (2025, May 13). *Recovering Snapchats: Law enforcement's digital forensics challenge.* Lawshun.com.

Office of the Privacy Commissioner of Canada. (2024, May 1). *PIPEDA in brief.* https://www.priv.gc.ca

Perloff, R. M. (1999). The third-person effect: A critical review and synthesis. *Media Psychology, 1*(4), 353–378.

Persson, P. (2023, June 7). *Digital forensic analysis of Snapchat and BeReal* [Bachelor's thesis, Halmstad University]. DiVA Portal.

Rashid, E., & Mastorakis, N. (2025). Elimination and analysis of ephemeral messages in Android social media apps: A forensic perspective. *IARAS.*

Sankuratripati, S., Yung, M., Garg, A., & Huang, W. (2019, January 9–11). *Catch me if you can: An account based end-to-end encryption for 1/1 snaps* [Conference presentation slides]. Real World Crypto 2019, San Jose, CA, United States. https://rwc.iacr.org/2019/slides/snap.pdf

Snap Inc. (2021, June 24). *QUIC at Snapchat.* Snap Engineering. https://eng.snap.com/quic-at-snap

Snapchat. (n.d.). *How do I add or remove login information from my device?* Snapchat Support. https://help.snapchat.com/hc/en-us/articles/39838995368724-How-do-I-add-or-remove-login-information-from-my-device

Sundar, S. S. (2008). The MAIN model: A heuristic approach to understanding technology effects on credibility. In M. J. Metzger & A. J. Flanagin (Eds.), *Digital media, youth, and credibility* (pp. 73–100). MIT Press.

TLS certificate pinning 101. (2018, March 13). LRQA. https://www.lrqa.com/en/cyber-labs/tls-certificate-pinning-101/

Van Essen, T., & Van Ouytsel, J. (2023). Snapchat streaks and problematic smartphone use. *Cyberpsychology, Behavior, and Social Networking.*

Waddell, T. F. (2016). The allure of privacy or the desire for self-expression? *Cyberpsychology, Behavior, and Social Networking, 19*(7), 441–445.

Appendices

Appendix A: Device Preparation and Rooting Procedures

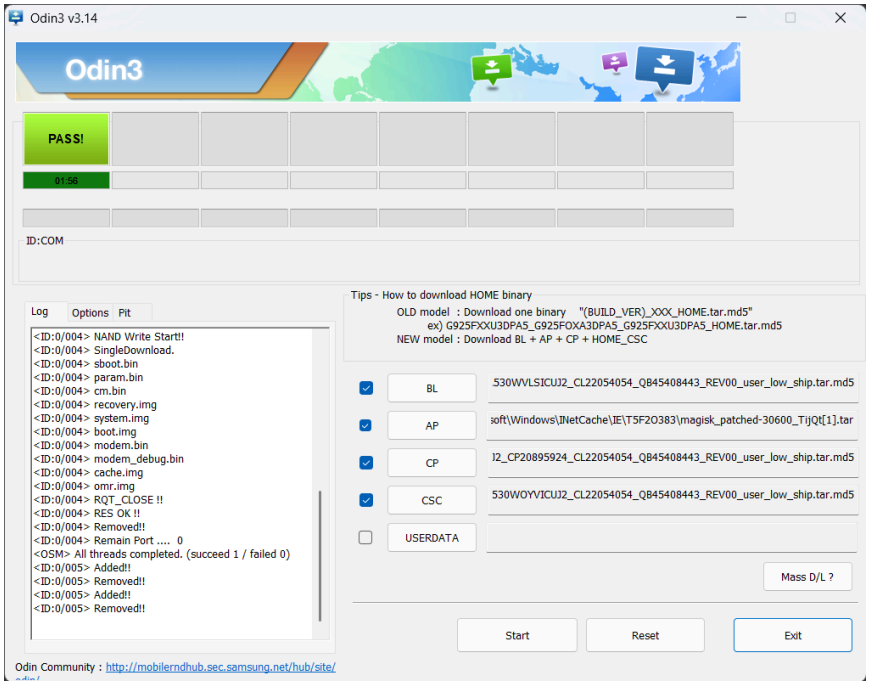


Figure A1. Odin3 software showing successful firmware flash using Magisk-Patched AP file.

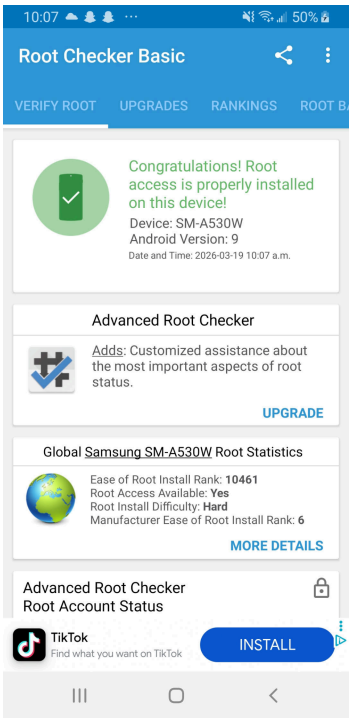


Figure A2. Root state verified through Root Checker app.

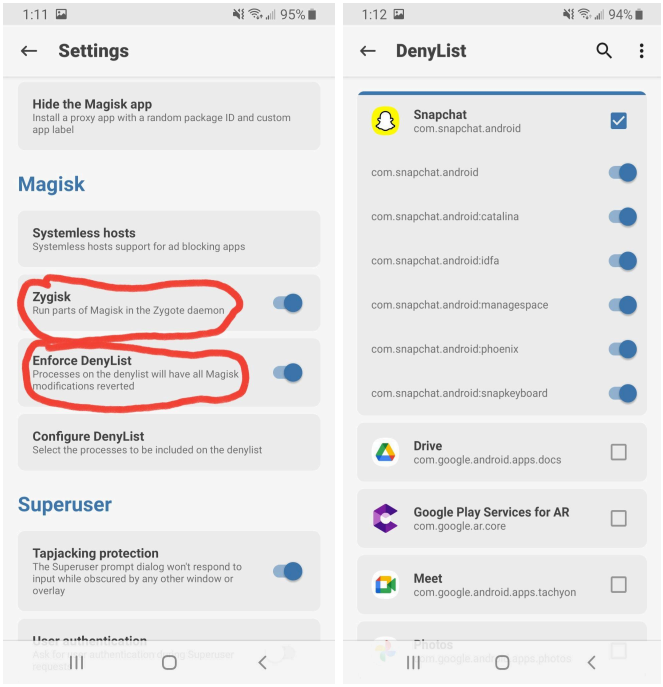


Figure A3. Zygisk enabled and DenyList configured for Snapchat root detection bypass on Galaxy A8.

Appendix B: Network Infrastructure Configuration

```
# Interface
interface=ap0
driver=nl80211


# Network
ssid=SnapForensics
hw_mode=g
channel=11
wmm_enabled=0


# 802.11n support
ieee80211n=1

# Authentication
macaddr_acl=0
auth_algs=1
ignore_broadcast_ssid=0


# Security
wpa=2
wpa_passphrase=KaleysForensicLab01307337!
wpa_key_mgmt=WPA-PSK
wpa_pairwise=TKIP
rsn_pairwise=CCMP


# Logging
logger_syslog=-1
logger_syslog_level=2
logger_stdout=-1
logger_stdout_level=2
```

Figure B1. Hostapd configuration file showing the software access point setup with channel and SSID parameters.

```
interface=ap0
bind-interfaces


dhcp-range=192.168.100.10,192.168.100.100,24h


# Gateway (laptop)
dhcp-option=3,192.168.100.1


# DNS Server
dhcp-option=6,8.8.8.8,8.8.4.4


# Lease file
dhcp-leasefile=/var/lib/misc/dnsmasq.leases


# Logging
log-queries
log-dhcp
```

Figure B2. Dnsmasq configuration for DHCP services on the isolated forensic network.

```
(mitmproxy-env) kn@kn-tp:~/.mitmproxy$ python3 -m http.server 8888 --bind 192.168.1.142
Serving HTTP on 192.168.1.142 port 8888 (http://192.168.1.142:8888/) ...
192.168.1.92 - - [21/Jan/2026 20:04:03] "GET / HTTP/1.1" 200 -
192.168.1.92 - - [21/Jan/2026 20:04:03] code 404, message File not found
192.168.1.92 - - [21/Jan/2026 20:04:03] "GET /favicon.ico HTTP/1.1" 404 -
192.168.1.92 - - [21/Jan/2026 20:04:17] "GET /mitmproxy-ca-cert.pem HTTP/1.1" 200 -
192.168.1.191 - - [21/Jan/2026 20:07:24] "GET / HTTP/1.1" 200 -
192.168.1.191 - - [21/Jan/2026 20:07:25] code 404, message File not found
192.168.1.191 - - [21/Jan/2026 20:07:25] "GET /favicon.ico HTTP/1.1" 404 -
192.168.1.191 - - [21/Jan/2026 20:07:47] "GET /mitmproxy-ca-cert.pem HTTP/1.1" 200 -
```

Figure B3. Local HTTP server hosting for mitmproxy certificate distribution.

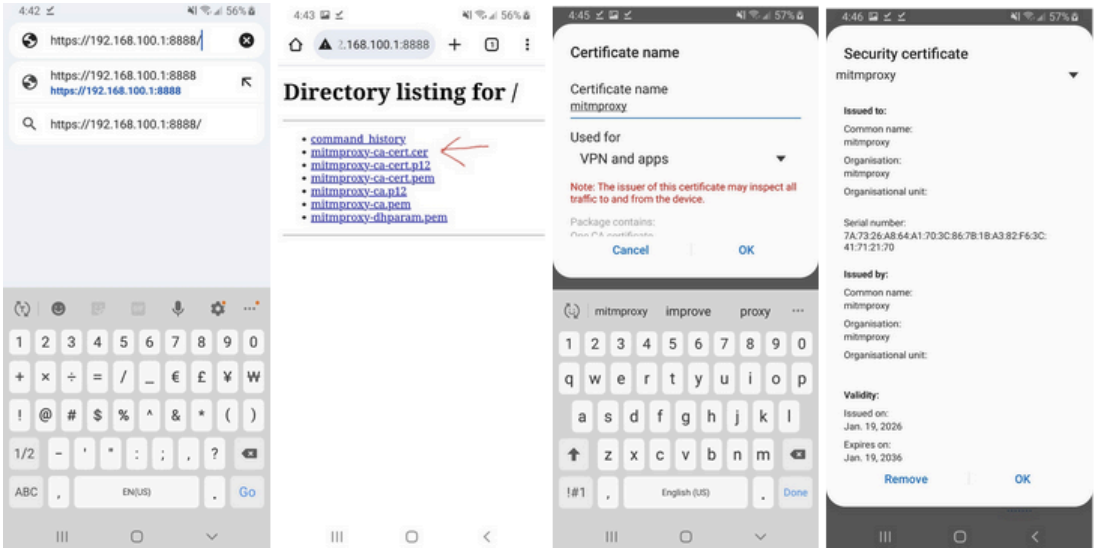


Figure B4. Certificate installation process on the Samsung Galaxy A8

```
(mitmproxy-env) kn@kn-tp:~/Desktop$ adb shell su -c "/data/local/tmp/frida-server &"
* daemon not running; starting now at tcp:5037
* daemon started successfully
/system/bin/sh: /data/local/tmp/frida-server: not found
(mitmproxy-env) kn@kn-tp:~/Desktop$ frida-ps -U
```

Figure B5. Frida server initialization and connection verification on the Samsung Galaxy A8.

Appendix C: Network Capture Session

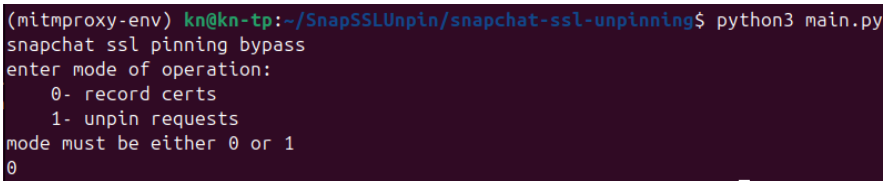


Figure C1. Initial capture of Snapchat domain requests

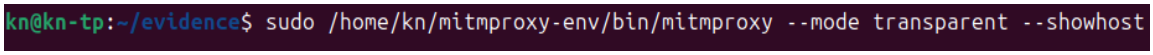


Figure C2. Mitmproxy instance initialization

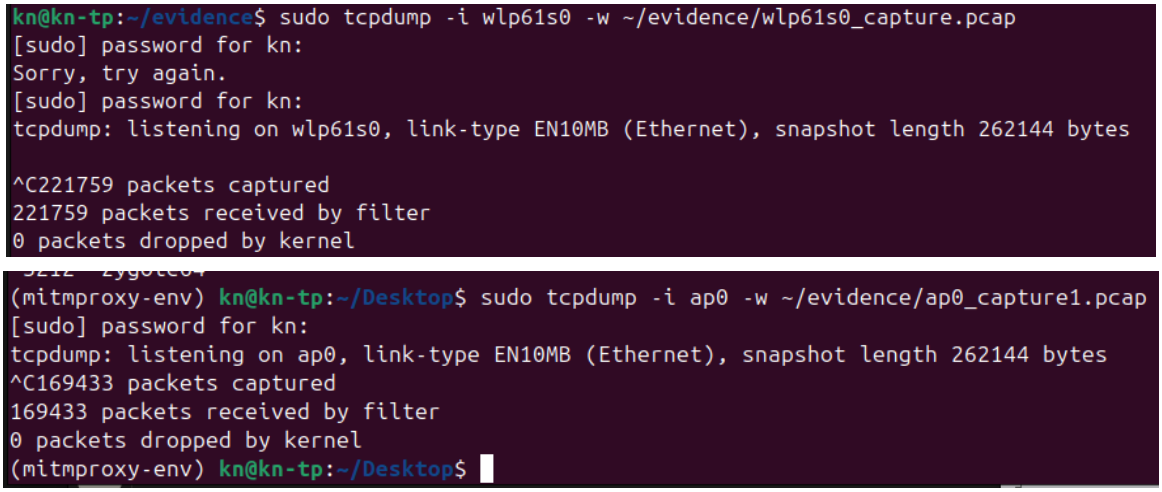


Figure C3. Tcpdump packet captures on ap0 and wlp61s0 initialization

Appendix D: Device Forensic Acquisition Process

```
PS C:\platform-tools> .\adb shell
jackpotltecan:/ $ su
jackpotltecan:/ # whoami
root
jackpotltecan:/ #
```

Figure D1. Opening adb shell and confirming root.

```
/dev/block/mmcblk0p19 on /system/lib64/vendor.samsung.hardware.camera.device@1.0.so type ext4 (ro,seclabel,relatime)
/dev/block/mmcblk0p19 on /system/lib64/vendor.samsung.hardware.camera.device@3.2.so type ext4 (ro,seclabel,relatime)
/dev/block/mmcblk0p19 on /system/lib64/vendor.samsung.hardware.camera.device@3.3.so type ext4 (ro,seclabel,relatime)
/dev/block/mmcblk0p19 on /system/lib64/vendor.samsung.hardware.camera.device@3.4.so type ext4 (ro,seclabel,relatime)
/dev/block/mmcblk0p19 on /system/lib64/vendor.samsung.hardware.camera.provider@2.4.so type ext4 (ro,seclabel,relatime)
/dev/block/mmcblk0p19 on /system/lib64/vendor.samsung.hardware.exthealth@1.0.so type ext4 (ro,seclabel,relatime)
/dev/block/mmcblk0p19 on /system/lib64/vendor.samsung.hardware.gnss@1.0.so type ext4 (ro,seclabel,relatime)
/dev/block/mmcblk0p19 on /system/lib64/vendor.samsung.hardware.light@2.0.so type ext4 (ro,seclabel,relatime)
/dev/block/mmcblk0p19 on /system/lib64/vendor.samsung.hardware.miscpower@1.0.so type ext4 (ro,seclabel,relatime)
/dev/block/mmcblk0p19 on /system/lib64/vendor.samsung.hardware.nfc@1.1.so type ext4 (ro,seclabel,relatime)
/dev/block/mmcblk0p19 on /system/lib64/vendor.samsung.hardware.radio.secbridge@1.0.so type ext4 (ro,seclabel,relatime)
/dev/block/mmcblk0p19 on /system/lib64/vendor.samsung.hardware.vibrator@1.0.so type ext4 (ro,seclabel,relatime)
/dev/block/mmcblk0p19 on /system/lib64/vendor.samsung.hardware.wifi.hostapd@1.0.so type ext4 (ro,seclabel,relatime)
/dev/block/mmcblk0p19 on /system/lib64/vendor.samsung.hardware.wifi.supplicant@1.0.so type ext4 (ro,seclabel,relatime)
/dev/block/mmcblk0p19 on /system/lib64/vendor.samsung.hardware.wifi@1.0.so type ext4 (ro,seclabel,relatime)
/dev/block/mmcblk0p19 on /system/lib64/vendor.samsung.security.proca@1.0.so type ext4 (ro,seclabel,relatime)
/dev/block/mmcblk0p19 on /system/lib64/vendor.samsung.security.sem@1.0.so type ext4 (ro,seclabel,relatime)
/dev/block/mmcblk0p19 on /system/lib64/vendor.samsung.security.skpm@1.0.so type ext4 (ro,seclabel,relatime)
/dev/block/mmcblk0p19 on /system/lib64/vendor.samsung.security.vaultkeeper_server@1.0.so type ext4 (ro,seclabel,relatime)
/dev/block/mmcblk0p19 on /system/lib64/vendor.samsung.security.wvkprov_server@1.0.so type ext4 (ro,seclabel,relatime)
/dev/block/mmcblk0p19 on /system/lib64/vendor.samsung_slsi.hardware.ExynosHWCServiceTW@1.0.so type ext4 (ro,seclabel,relatime)
/dev/block/mmcblk0p19 on /system/lib64/vendor.samsung_slsi.hardware.configstore-utils.so type ext4 (ro,seclabel,relatime)
/dev/block/mmcblk0p19 on /system/lib64/vendor.samsung_slsi.hardware.configstore@1.0.so type ext4 (ro,seclabel,relatime)
/dev/block/mmcblk0p19 on /system/lib64/vendor.trustonic.tee@1.0.so type ext4 (ro,seclabel,relatime)
/dev/block/mmcblk0p19 on /system/lib64/vendor.trustonic.teeregistry@1.0.so type ext4 (ro,seclabel,relatime)
magisk on /system/lib64/libzygisk.so type tmpfs (ro,seclabel,relatime,size=1886268k,nr_inodes=471567,mode=755)
/data/knox/secure_fs/enc_user on /data/enc_user type ecryptfs (rw,seclabel,nodev,relatime,ecryptfs_fnek_sig=1695a5beb0c62ed0,ecryptfs_sig=1695a5beb0c62ed0,u
serid=0,sdp_enabled,partition_id=0,ecryptfs_cipher=aes,ecryptfs_key_bytes=32,ecryptfs_passthrough,base=,label=)
/data/knox/secure_fs/enc_media on /data/knox/secure_fs/enc_media type ecryptfs (rw,seclabel,nodev,relatime,ecryptfs_fnek_sig=1695a5beb0c62ed0,ecryptfs_sig=1
695a5beb0c62ed0,userid=0,sdp_enabled,partition_id=1,ecryptfs_cipher=aes,ecryptfs_key_bytes=32,ecryptfs_passthrough,base=,label=)
/data/knox/secure_fs/enc_media on /mnt/shell/enc_emulated type sdcardfs (rw,nosuid,nodev,noexec,noatime,fsuid=1000,fsgid=1000,gid=9997,multiuser,derive_gid,
default_normal,reserved=20MB)
/data/media on /mnt/runtime/default/emulated type sdcardfs (rw,nosuid,nodev,noexec,noatime,fsuid=1023,fsgid=1023,gid=1015,multiuser,mask=6,derive_gid,defaul
t_normal,reserved=20MB)
/data/media on /storage/emulated type sdcardfs (rw,nosuid,nodev,noexec,noatime,fsuid=1023,fsgid=1023,gid=1015,multiuser,mask=6,derive_gid,default_normal,res
erved=20MB)
/data/media on /mnt/runtime/read/emulated type sdcardfs (rw,nosuid,nodev,noexec,noatime,fsuid=1023,fsgid=1023,gid=9997,multiuser,mask=23,derive_gid,default_
normal,reserved=20MB)
/data/media on /mnt/runtime/write/emulated type sdcardfs (rw,nosuid,nodev,noexec,noatime,fsuid=1023,fsgid=1023,gid=9997,multiuser,mask=7,derive_gid,default_
normal,reserved=20MB)
jackpotltecan:/ # mount
```

Figure D2. Galaxy A8 partition being mounted to prepare for archiving.

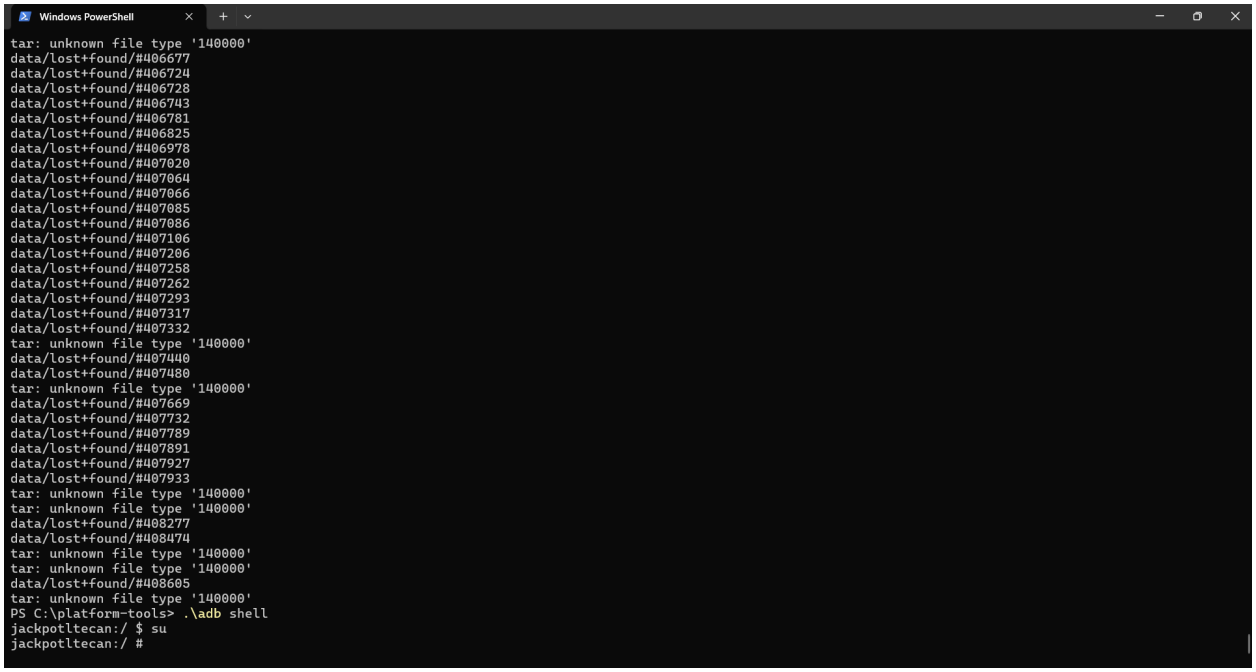


Figure D3. /data partition mounted and successful result of Galaxy A8 archive creation.

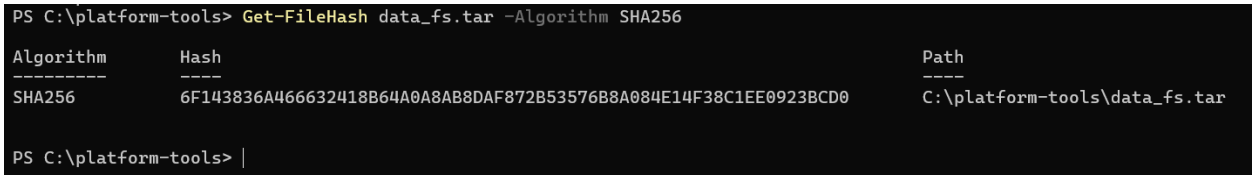


Figure D4. Hash value verification

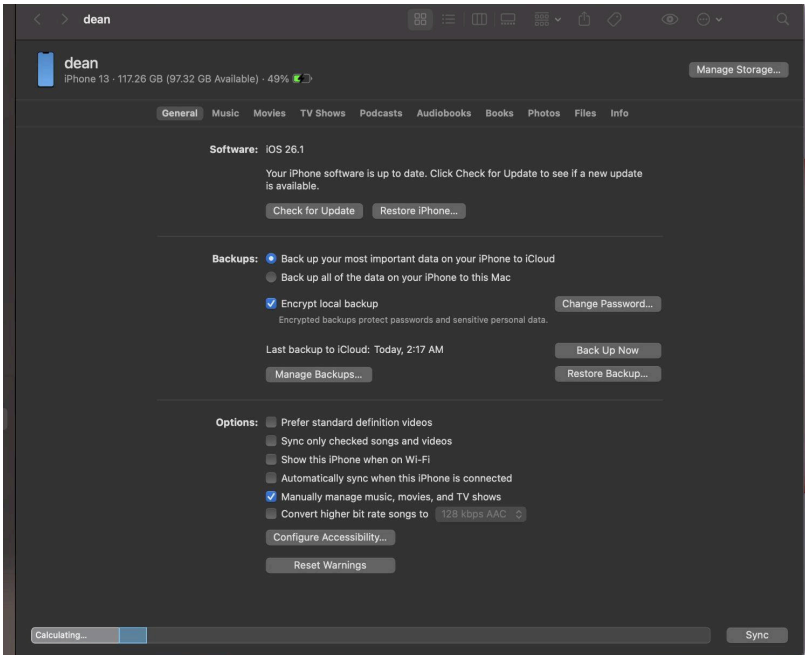


Figure D5. iPhone 13 connected to iTunes for an encrypted backup.

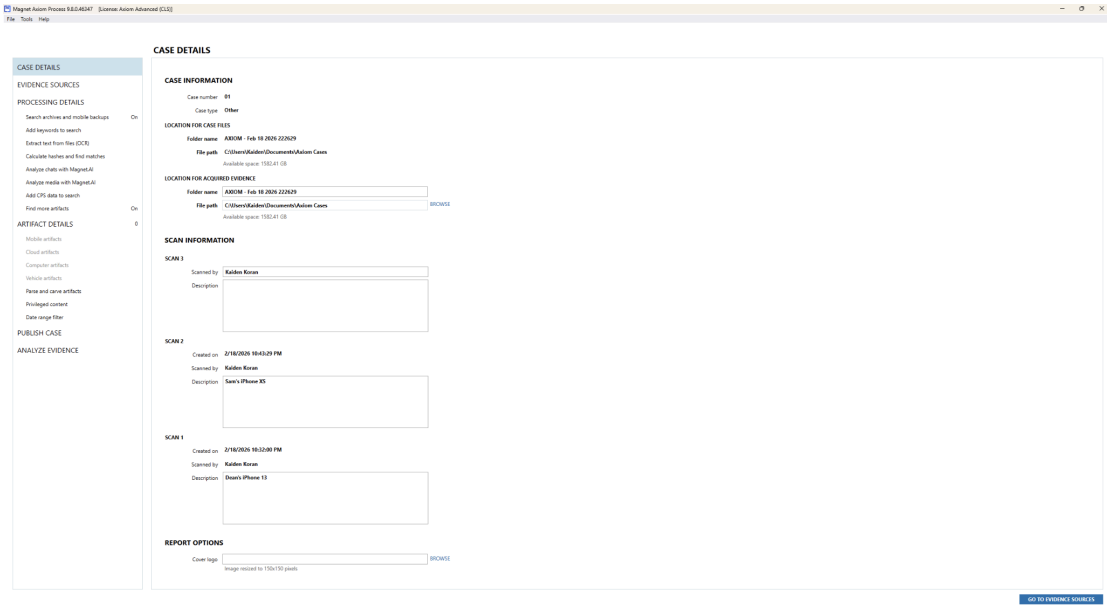


Figure D6. Magnet Axiom case creation with both backups

Appendix E: Additional Network Traffic Analysis Captures

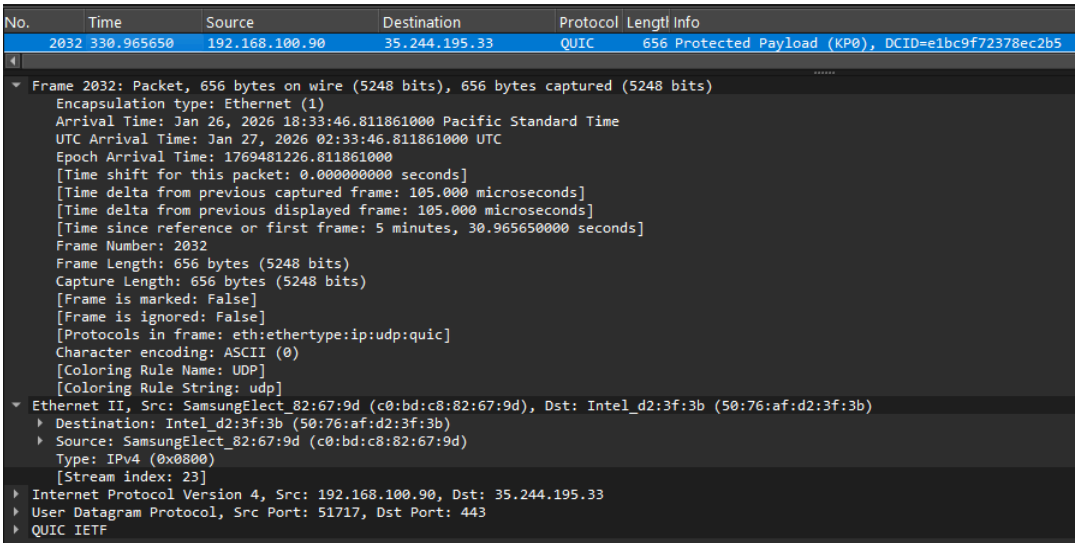


Figure E1. Continuous communication pattern for Galaxy A8 in Wireshark, upload packet.

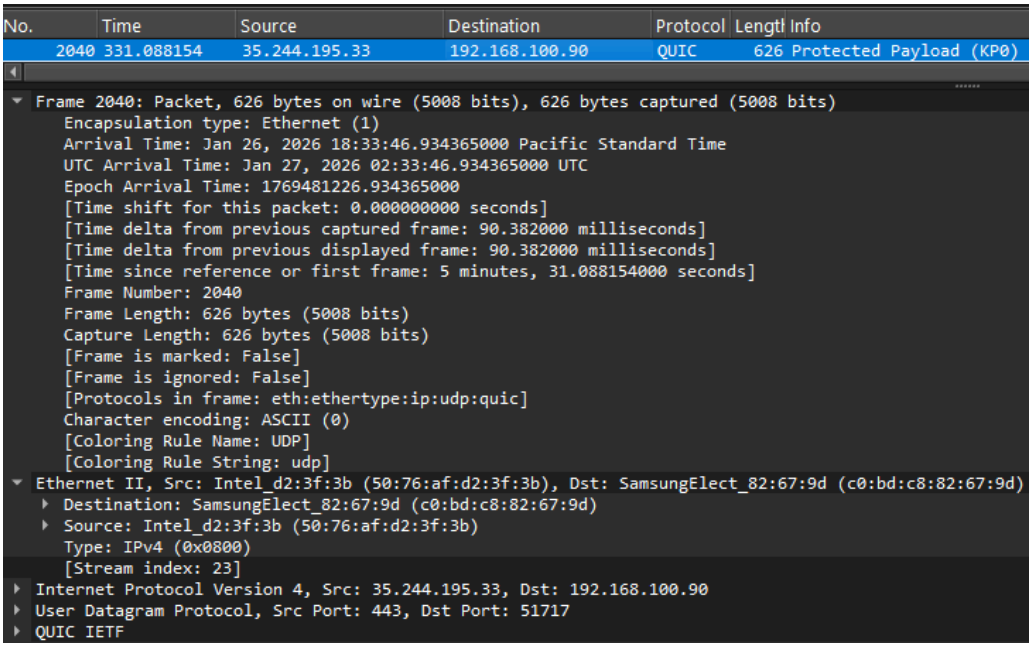


Figure E2. Continuous communication pattern with Galaxy A8 in Wireshark, response packet.

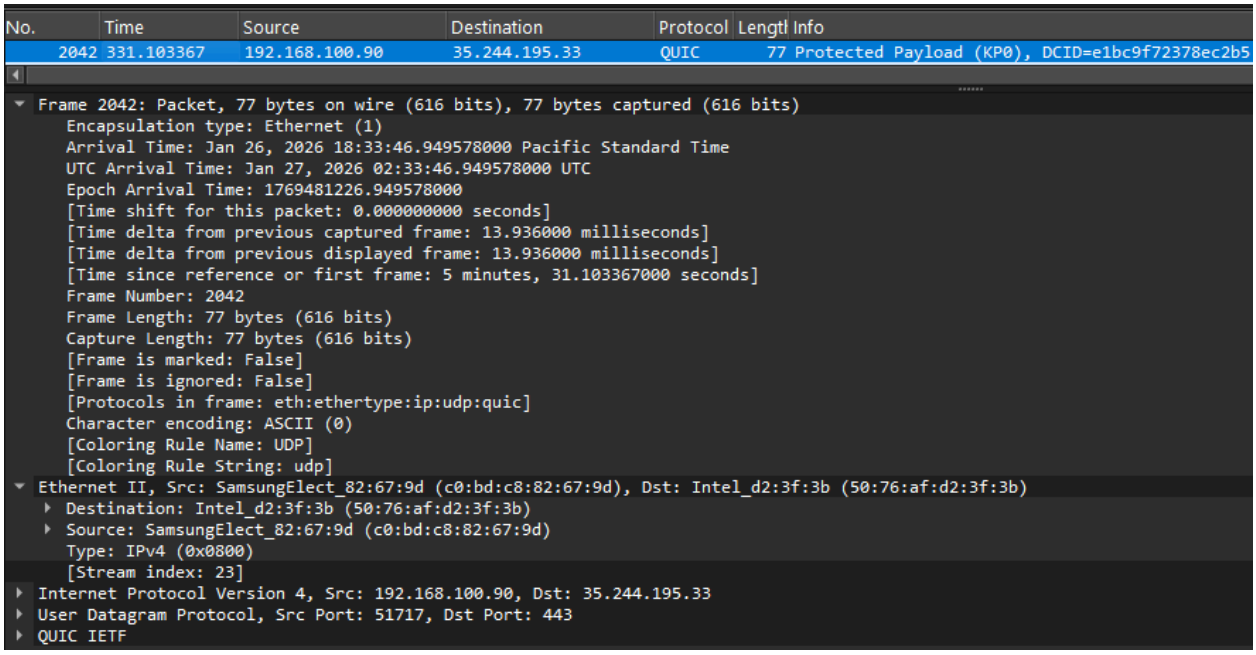


Figure E3. Continuous communication pattern with Galaxy A8 in Wireshark device acknowledgement packet.

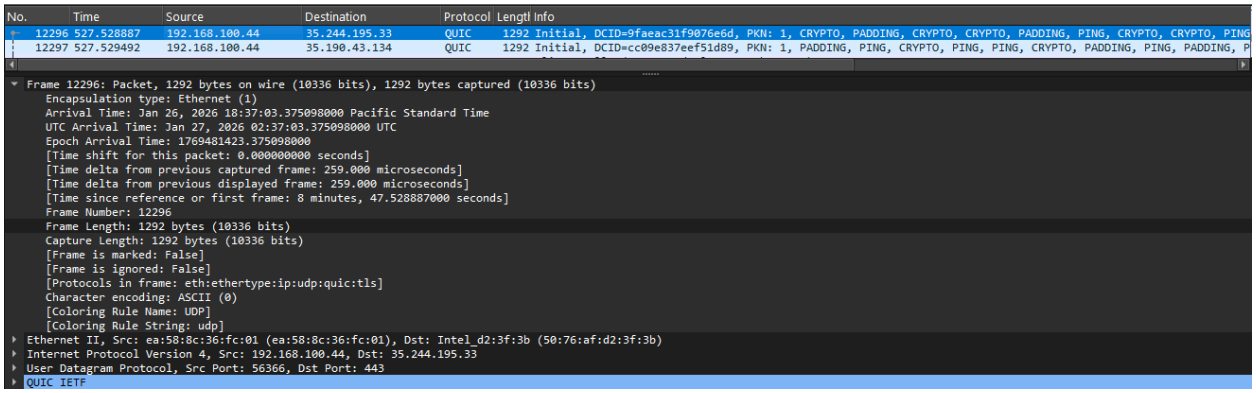


Figure E4. iPhone XS first QUIC contact with Snapchat servers.

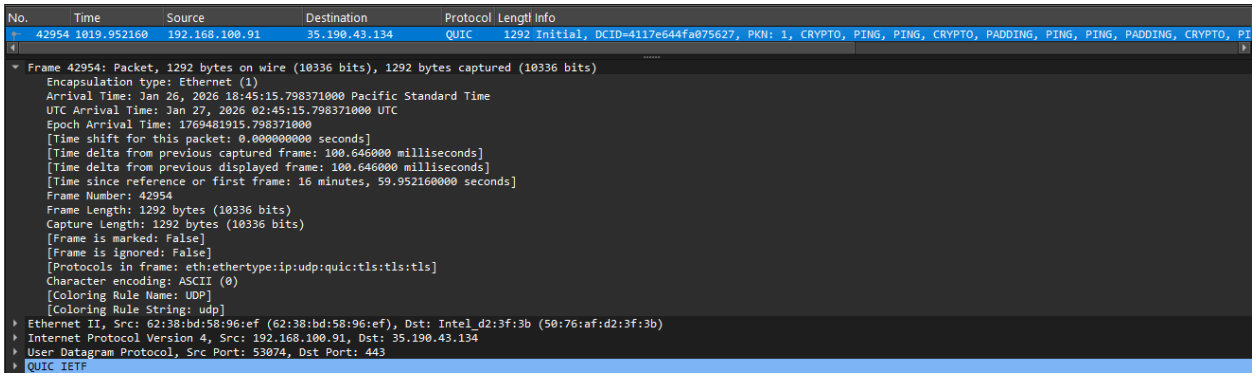
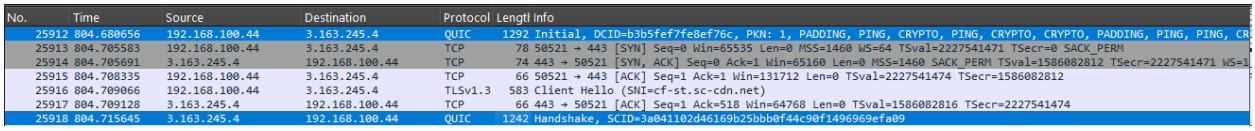
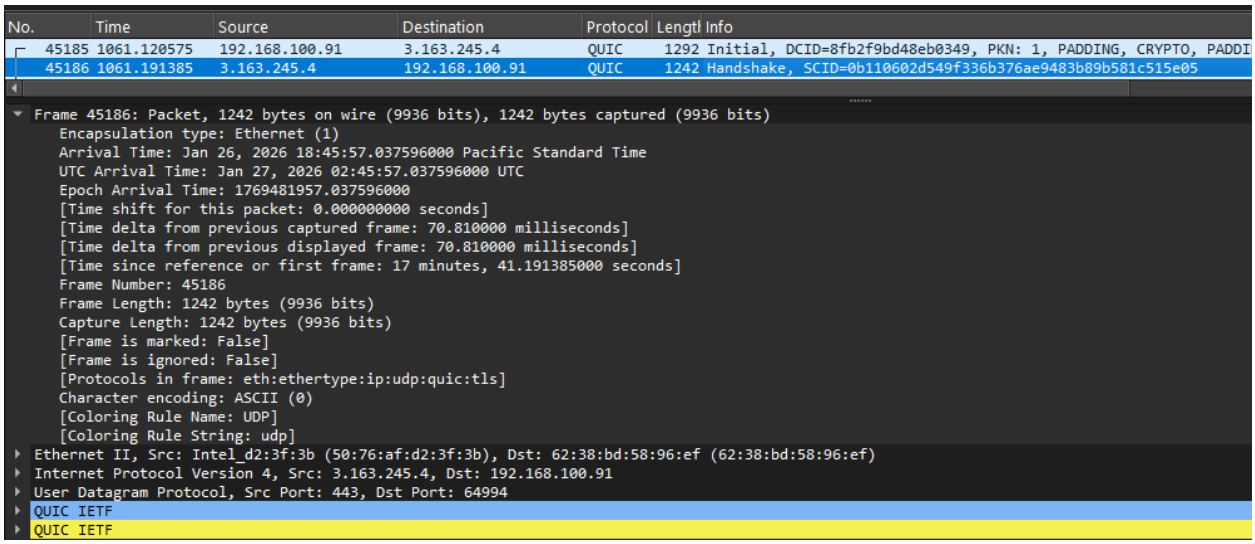


Figure E5. First QUIC contact on iPhone 13.



No.	Time	Source	Destination	Protocol	Lengtl	Info
25912	804.680656	192.168.100.44	3.163.245.4	QUIC	1292	Initial, DCID=b3b5fef7fe8ef76c, PKN: 1, PADDING, PING, CRYPTO, PING, CRYPTO, CRYPTO, PADDING, PING, PING, CR
25913	804.705583	192.168.100.44	3.163.245.4	TCP	78	50521 → 443 [SYN] Seq=0 Win=65535 Len=0 MSS=1460 WS=64 TSval=2227541471 TSecr=0 SACK_PERM
25914	804.705691	3.163.245.4	192.168.100.44	TCP	74	443 → 50521 [SYN, ACK] Seq=0 Ack=1 Win=65160 Len=0 MSS=1460 SACK_PERM TSval=1586082812 TSecr=2227541471 WS=1
25915	804.708335	192.168.100.44	3.163.245.4	TCP	66	50521 → 443 [ACK] Seq=1 Ack=1 Win=131712 Len=0 TSval=2227541474 TSecr=1586082812
25916	804.709066	192.168.100.44	3.163.245.4	TLSv1.3	583	Client Hello (SNI=cf-st.sc-cdn.net)
25917	804.709128	3.163.245.4	192.168.100.44	TCP	66	443 → 50521 [ACK] Seq=1 Ack=518 Win=64768 Len=0 TSval=1586082816 TSecr=2227541474
25918	804.715645	3.163.245.4	192.168.100.44	QUIC	1242	Handshake, SCID=3a041102d46169b25bbb0f44c90f1496969efa09

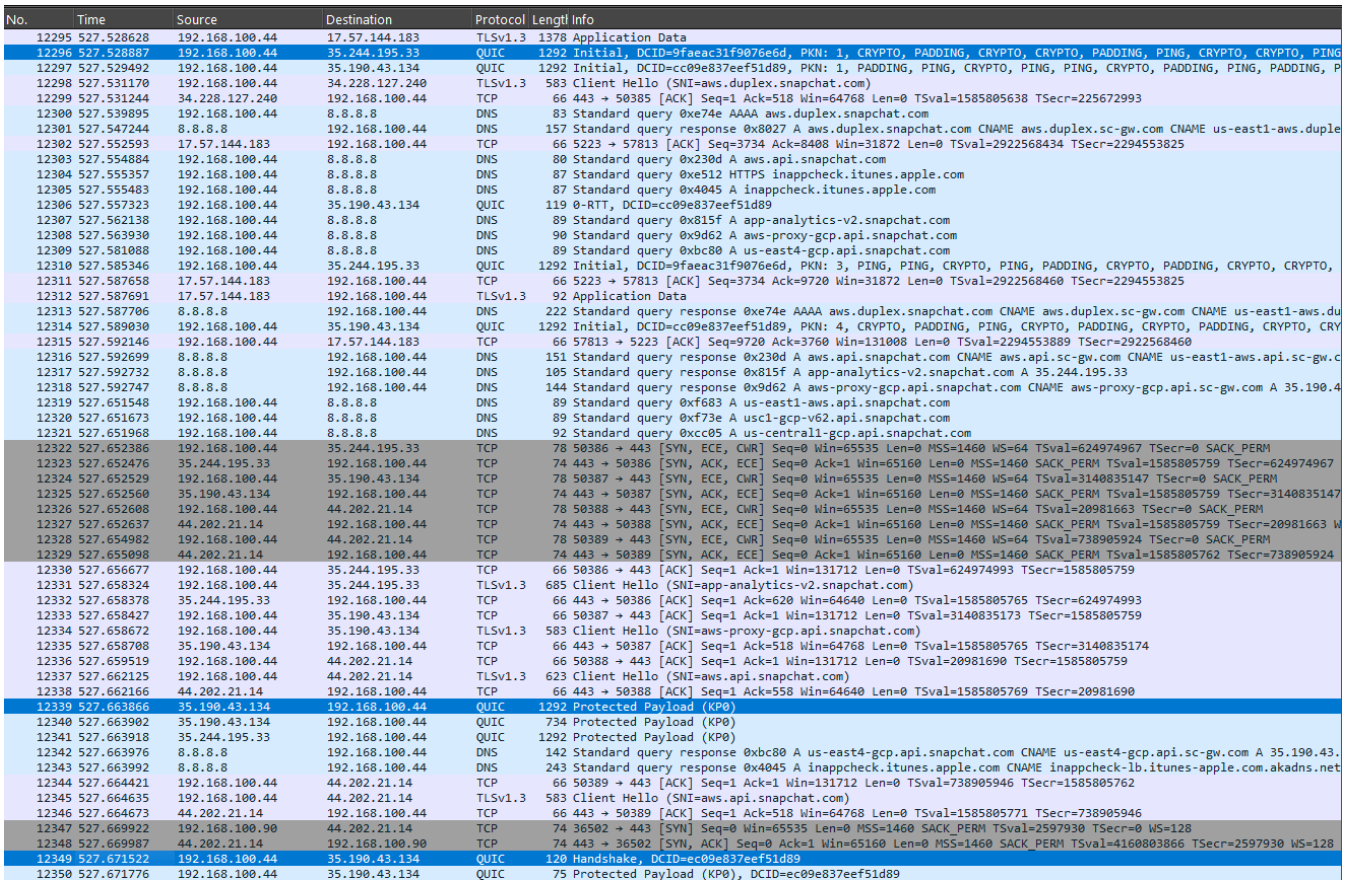
Figure E6. iPhone XS's first contact with AWS media server.



No.	Time	Source	Destination	Protocol	Lengtl	Info
45185	1061.120575	192.168.100.91	3.163.245.4	QUIC	1292	Initial, DCID=8fb2f9bd48eb0349, PKN: 1, PADDING, CRYPTO, PADDI
45186	1061.191385	3.163.245.4	192.168.100.91	QUIC	1242	Handshake, SCID=0b110602d549f336b376ae9483b89b581c515e05

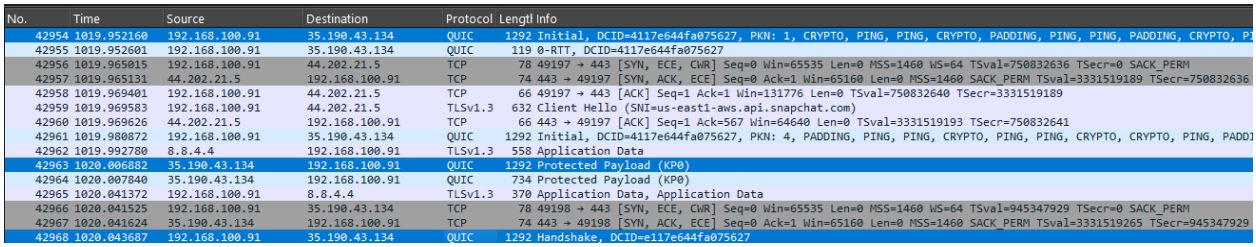
```
Frame 45186: Packet, 1242 bytes on wire (9936 bits), 1242 bytes captured (9936 bits)
    Encapsulation type: Ethernet (1)
    Arrival Time: Jan 26, 2026 18:45:57.037596000 Pacific Standard Time
    UTC Arrival Time: Jan 27, 2026 02:45:57.037596000 UTC
    Epoch Arrival Time: 1769481957.037596000
    [Time shift for this packet: 0.000000000 seconds]
    [Time delta from previous captured frame: 70.810000 milliseconds]
    [Time delta from previous displayed frame: 70.810000 milliseconds]
    [Time since reference or first frame: 17 minutes, 41.191385000 seconds]
    Frame Number: 45186
    Frame Length: 1242 bytes (9936 bits)
    Capture Length: 1242 bytes (9936 bits)
    [Frame is marked: False]
    [Frame is ignored: False]
    [Protocols in frame: eth:ethertype:ip:udp:quic:tls]
    Character encoding: ASCII (0)
    [Coloring Rule Name: UDP]
    [Coloring Rule String: udp]
▸ Ethernet II, Src: Intel_d2:3f:3b (50:76:af:d2:3f:3b), Dst: 62:38:bd:58:96:ef (62:38:bd:58:96:ef)
▸ Internet Protocol Version 4, Src: 3.163.245.4, Dst: 192.168.100.91
▸ User Datagram Protocol, Src Port: 443, Dst Port: 64994
▸ QUIC IETF
▸ QUIC IETF
```

Figure E7. iPhone 13 first contact with AWS server.



No.	Time	Source	Destination	Protocol	Lengtl	Info
12295	527.528628	192.168.100.44	17.57.144.183	TLSv1.3	1378	Application Data
12296	527.528887	192.168.100.44	35.244.195.33	QUIC	1292	Initial, DCID=9faeac31f9076e6d, PKN: 1, CRYPTO, PADDING, CRYPTO, CRYPTO, PADDING, PING, CRYPTO, CRYPTO, PING
12297	527.529492	192.168.100.44	35.190.43.134	QUIC	1292	Initial, DCID=cc09e837eef51d89, PKN: 1, PADDING, PING, CRYPTO, PING, PING, CRYPTO, PADDING, PING, PADDING, P
12298	527.531170	192.168.100.44	34.228.127.240	TLSv1.3	583	Client Hello (SNI=aws.duplex.snapchat.com)
12299	527.531244	34.228.127.240	192.168.100.44	TCP	66	443 → 50385 [ACK] Seq=1 Ack=518 Win=64768 Len=0 TSval=1585805638 TSecr=225672993
12300	527.539895	192.168.100.44	8.8.8.8	DNS	83	Standard query 0xe74e AAAA aws.duplex.snapchat.com
12301	527.547244	8.8.8.8	192.168.100.44	DNS	157	Standard query response 0x8027 A aws.duplex.snapchat.com CNAME aws.duplex.sc-gw.com CNAME us-east1-aws.duple
12302	527.552593	17.57.144.183	192.168.100.44	TCP	66	5223 → 57813 [ACK] Seq=3734 Ack=8408 Win=31872 Len=0 TSval=2922568434 TSecr=2294553825
12303	527.554884	192.168.100.44	8.8.8.8	DNS	80	Standard query 0x230d A aws.api.snapchat.com
12304	527.555357	192.168.100.44	8.8.8.8	DNS	87	Standard query 0xe512 HTTPS inappcheck.itunes.apple.com
12305	527.555483	192.168.100.44	8.8.8.8	DNS	87	Standard query 0x4045 A inappcheck.itunes.apple.com
12306	527.557323	192.168.100.44	35.190.43.134	QUIC	119	0-RTT, DCID=cc09e837eef51d89
12307	527.562138	192.168.100.44	8.8.8.8	DNS	89	Standard query 0x815f A app-analytics-v2.snapchat.com
12308	527.563930	192.168.100.44	8.8.8.8	DNS	90	Standard query 0x9d62 A aws-proxy-gcp.api.snapchat.com
12309	527.581088	192.168.100.44	8.8.8.8	DNS	89	Standard query 0xbc80 A us-east4-gcp.api.snapchat.com
12310	527.585346	192.168.100.44	35.244.195.33	QUIC	1292	Initial, DCID=9faeac31f9076e6d, PKN: 3, PING, PING, CRYPTO, PING, PADDING, CRYPTO, PADDING, CRYPTO, CRYPTO,
12311	527.587658	17.57.144.183	192.168.100.44	TCP	66	5223 → 57813 [ACK] Seq=3734 Ack=9720 Win=31872 Len=0 TSval=2922568460 TSecr=2294553825
12312	527.587691	17.57.144.183	192.168.100.44	TLSv1.3	92	Application Data
12313	527.587706	8.8.8.8	192.168.100.44	DNS	222	Standard query response 0xe74e AAAA aws.duplex.snapchat.com CNAME aws.duplex.sc-gw.com CNAME us-east1-aws.du
12314	527.589030	192.168.100.44	35.190.43.134	QUIC	1292	Initial, DCID=cc09e837eef51d89, PKN: 4, CRYPTO, PADDING, PING, CRYPTO, PADDING, CRYPTO, PADDING, CRYPTO, CRY
12315	527.592146	192.168.100.44	17.57.144.183	TCP	66	57813 → 5223 [ACK] Seq=9720 Ack=3760 Win=131008 Len=0 TSval=2294553889 TSecr=2922568460
12316	527.592699	8.8.8.8	192.168.100.44	DNS	151	Standard query response 0x230d A aws.api.snapchat.com CNAME aws.api.sc-gw.com CNAME us-east1-aws.api.sc-gw.c
12317	527.592732	8.8.8.8	192.168.100.44	DNS	105	Standard query response 0x815f A app-analytics-v2.snapchat.com A 35.244.195.33
12318	527.592747	8.8.8.8	192.168.100.44	DNS	144	Standard query response 0x9d62 A aws-proxy-gcp.api.snapchat.com CNAME aws-proxy-gcp.api.sc-gw.com A 35.190.4
12319	527.651548	192.168.100.44	8.8.8.8	DNS	89	Standard query 0xf683 A us-east1-aws.api.snapchat.com
12320	527.651673	192.168.100.44	8.8.8.8	DNS	89	Standard query 0xf73e A usc1-gcp-v62.api.snapchat.com
12321	527.651968	192.168.100.44	8.8.8.8	DNS	92	Standard query 0xcc05 A us-central1-gcp.api.snapchat.com
12322	527.652386	192.168.100.44	35.244.195.33	TCP	78	50386 → 443 [SYN, ECE, CWR] Seq=0 Win=65535 Len=0 MSS=1460 WS=64 TSval=624974967 TSecr=0 SACK_PERM
12323	527.652476	35.244.195.33	192.168.100.44	TCP	74	443 → 50386 [SYN, ACK, ECE] Seq=0 Ack=1 Win=65160 Len=0 MSS=1460 SACK_PERM TSval=1585805759 TSecr=624974967
12324	527.652529	192.168.100.44	35.190.43.134	TCP	78	50387 → 443 [SYN, ECE, CWR] Seq=0 Win=65535 Len=0 MSS=1460 WS=64 TSval=3140835147 TSecr=0 SACK_PERM
12325	527.652560	35.190.43.134	192.168.100.44	TCP	74	443 → 50387 [SYN, ACK, ECE] Seq=0 Ack=1 Win=65160 Len=0 MSS=1460 SACK_PERM TSval=1585805759 TSecr=3140835147
12326	527.652608	192.168.100.44	44.202.21.14	TCP	78	50388 → 443 [SYN, ECE, CWR] Seq=0 Win=65535 Len=0 MSS=1460 WS=64 TSval=20981663 TSecr=0 SACK_PERM
12327	527.652637	44.202.21.14	192.168.100.44	TCP	74	443 → 50388 [SYN, ACK, ECE] Seq=0 Ack=1 Win=65160 Len=0 MSS=1460 SACK_PERM TSval=1585805759 TSecr=20981663 W
12328	527.654982	192.168.100.44	44.202.21.14	TCP	78	50389 → 443 [SYN, ECE, CWR] Seq=0 Win=65535 Len=0 MSS=1460 WS=64 TSval=738905924 TSecr=0 SACK_PERM
12329	527.655098	44.202.21.14	192.168.100.44	TCP	74	443 → 50389 [SYN, ACK, ECE] Seq=0 Ack=1 Win=65160 Len=0 MSS=1460 SACK_PERM TSval=1585805762 TSecr=738905924
12330	527.656677	192.168.100.44	35.244.195.33	TCP	66	50386 → 443 [ACK] Seq=1 Ack=1 Win=131712 Len=0 TSval=624974993 TSecr=1585805759
12331	527.658324	192.168.100.44	35.244.195.33	TLSv1.3	685	Client Hello (SNI=app-analytics-v2.snapchat.com)
12332	527.658378	35.244.195.33	192.168.100.44	TCP	66	443 → 50386 [ACK] Seq=1 Ack=620 Win=64640 Len=0 TSval=1585805765 TSecr=624974993
12333	527.658427	192.168.100.44	35.190.43.134	TCP	66	50387 → 443 [ACK] Seq=1 Ack=1 Win=131712 Len=0 TSval=3140835173 TSecr=1585805759
12334	527.658672	192.168.100.44	35.190.43.134	TLSv1.3	583	Client Hello (SNI=aws-proxy-gcp.api.snapchat.com)
12335	527.658708	35.190.43.134	192.168.100.44	TCP	66	443 → 50387 [ACK] Seq=1 Ack=518 Win=64768 Len=0 TSval=1585805765 TSecr=3140835174
12336	527.659519	192.168.100.44	44.202.21.14	TCP	66	50388 → 443 [ACK] Seq=1 Ack=1 Win=131712 Len=0 TSval=20981690 TSecr=1585805759
12337	527.662125	192.168.100.44	44.202.21.14	TLSv1.3	623	Client Hello (SNI=aws.api.snapchat.com)
12338	527.662166	44.202.21.14	192.168.100.44	TCP	66	443 → 50388 [ACK] Seq=1 Ack=558 Win=64640 Len=0 TSval=1585805769 TSecr=20981690
12339	527.663866	35.190.43.134	192.168.100.44	QUIC	1292	Protected Payload (KP0)
12340	527.663902	35.190.43.134	192.168.100.44	QUIC	734	Protected Payload (KP0)
12341	527.663918	35.244.195.33	192.168.100.44	QUIC	1292	Protected Payload (KP0)
12342	527.663976	8.8.8.8	192.168.100.44	DNS	142	Standard query response 0xbc80 A us-east4-gcp.api.snapchat.com CNAME us-east4-gcp.api.sc-gw.com A 35.190.43.
12343	527.663992	8.8.8.8	192.168.100.44	DNS	243	Standard query response 0x4045 A inappcheck.itunes.apple.com CNAME inappcheck-lb.itunes-apple.com.akadns.net
12344	527.664421	192.168.100.44	44.202.21.14	TCP	66	50389 → 443 [ACK] Seq=1 Ack=1 Win=131712 Len=0 TSval=738905946 TSecr=1585805762
12345	527.664635	192.168.100.44	44.202.21.14	TLSv1.3	583	Client Hello (SNI=aws.api.snapchat.com)
12346	527.664673	44.202.21.14	192.168.100.44	TCP	66	443 → 50389 [ACK] Seq=1 Ack=518 Win=64768 Len=0 TSval=1585805771 TSecr=738905946
12347	527.669922	192.168.100.90	44.202.21.14	TCP	74	36502 → 443 [SYN] Seq=0 Win=65535 Len=0 MSS=1460 SACK_PERM TSval=2597930 TSecr=0 WS=128
12348	527.669987	44.202.21.14	192.168.100.90	TCP	74	443 → 36502 [SYN, ACK] Seq=0 Ack=1 Win=65160 Len=0 MSS=1460 SACK_PERM TSval=4160803866 TSecr=2597930 WS=128
12349	527.671522	192.168.100.44	35.190.43.134	QUIC	120	Handshake, DCID=ec09e837eef51d89
12350	527.671776	192.168.100.44	35.190.43.134	QUIC	75	Protected Payload (KP0), DCID=ec09e837eef51d89

Figure E8. Cross-communication with iPhone XS and Snapchat server



No.	Time	Source	Destination	Protocol	Lengtl	Info
42954	1019.952160	192.168.100.91	35.190.43.134	QUIC	1292	Initial, DCID=4117e644fa075627, PKN: 1, CRYPTO, PING, PING, CRYPTO, PADDING, PING, PING, PADDING, CRYPTO, PI
42955	1019.952601	192.168.100.91	35.190.43.134	QUIC	119	0-RTT, DCID=4117e644fa075627
42956	1019.965015	192.168.100.91	44.202.21.5	TCP	78	49197 → 443 [SYN, ECE, CWR] Seq=0 Win=65535 Len=0 MSS=1460 WS=64 TSval=750832636 TSecr=0 SACK_PERM
42957	1019.965131	44.202.21.5	192.168.100.91	TCP	74	443 → 49197 [SYN, ACK, ECE] Seq=0 Ack=1 Win=65160 Len=0 MSS=1460 SACK_PERM TSval=3331519189 TSecr=750832636
42958	1019.969401	192.168.100.91	44.202.21.5	TCP	66	49197 → 443 [ACK] Seq=1 Ack=1 Win=131776 Len=0 TSval=750832640 TSecr=3331519189
42959	1019.969583	192.168.100.91	44.202.21.5	TLSv1.3	632	Client Hello (SNI=us-east1-aws.api.snapchat.com)
42960	1019.969626	44.202.21.5	192.168.100.91	TCP	66	443 → 49197 [ACK] Seq=1 Ack=567 Win=64640 Len=0 TSval=3331519193 TSecr=750832641
42961	1019.980872	192.168.100.91	35.190.43.134	QUIC	1292	Initial, DCID=4117e644fa075627, PKN: 4, PADDING, PING, PING, CRYPTO, PING, PING, CRYPTO, CRYPTO, PING, PADDI
42962	1019.992780	8.8.4.4	192.168.100.91	TLSv1.3	558	Application Data
42963	1020.006882	35.190.43.134	192.168.100.91	QUIC	1292	Protected Payload (KP0)
42964	1020.007840	35.190.43.134	192.168.100.91	QUIC	734	Protected Payload (KP0)
42965	1020.041372	192.168.100.91	8.8.4.4	TLSv1.3	370	Application Data, Application Data
42966	1020.041525	192.168.100.91	35.190.43.134	TCP	78	49198 → 443 [SYN, ECE, CWR] Seq=0 Win=65535 Len=0 MSS=1460 WS=64 TSval=945347929 TSecr=0 SACK_PERM
42967	1020.041624	35.190.43.134	192.168.100.91	TCP	74	443 → 49198 [SYN, ACK, ECE] Seq=0 Ack=1 Win=65160 Len=0 MSS=1460 SACK_PERM TSval=3331519265 TSecr=945347929
42968	1020.043687	192.168.100.91	35.190.43.134	QUIC	1292	Handshake, DCID=e117e644fa075627

Figure E9. Cross communication packets with iPhone 13 and Snapchat server.

Appendix F: Samsung Galaxy A8 Artifact Screenshots

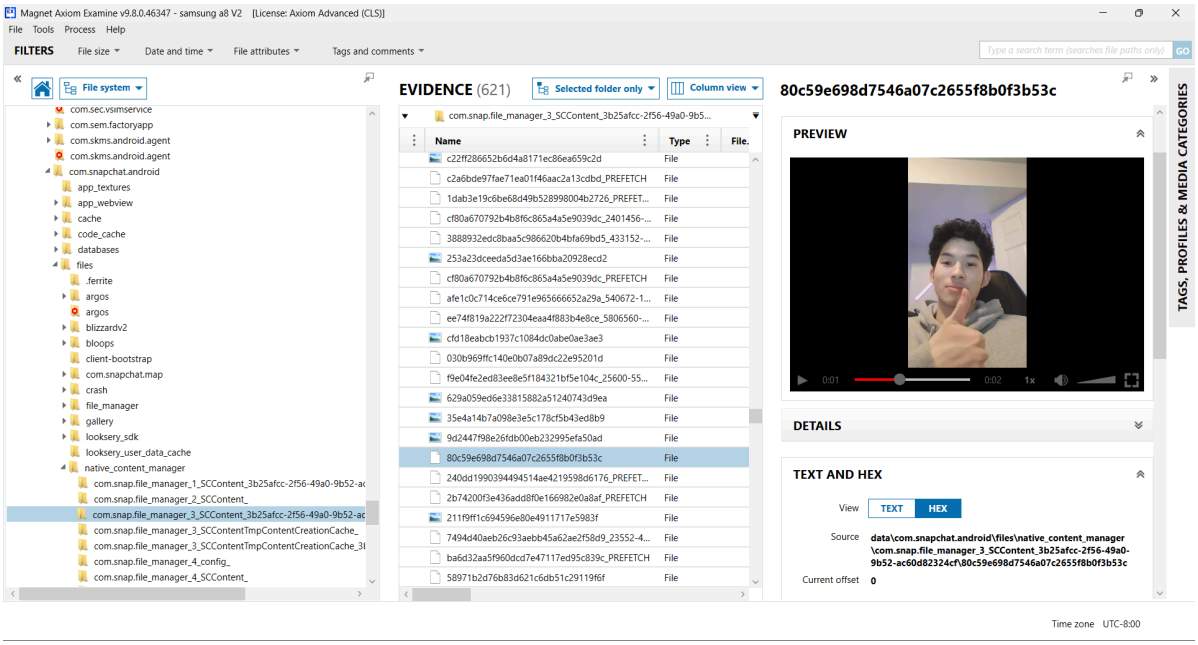


Figure F1. Video snap cache media recovered.

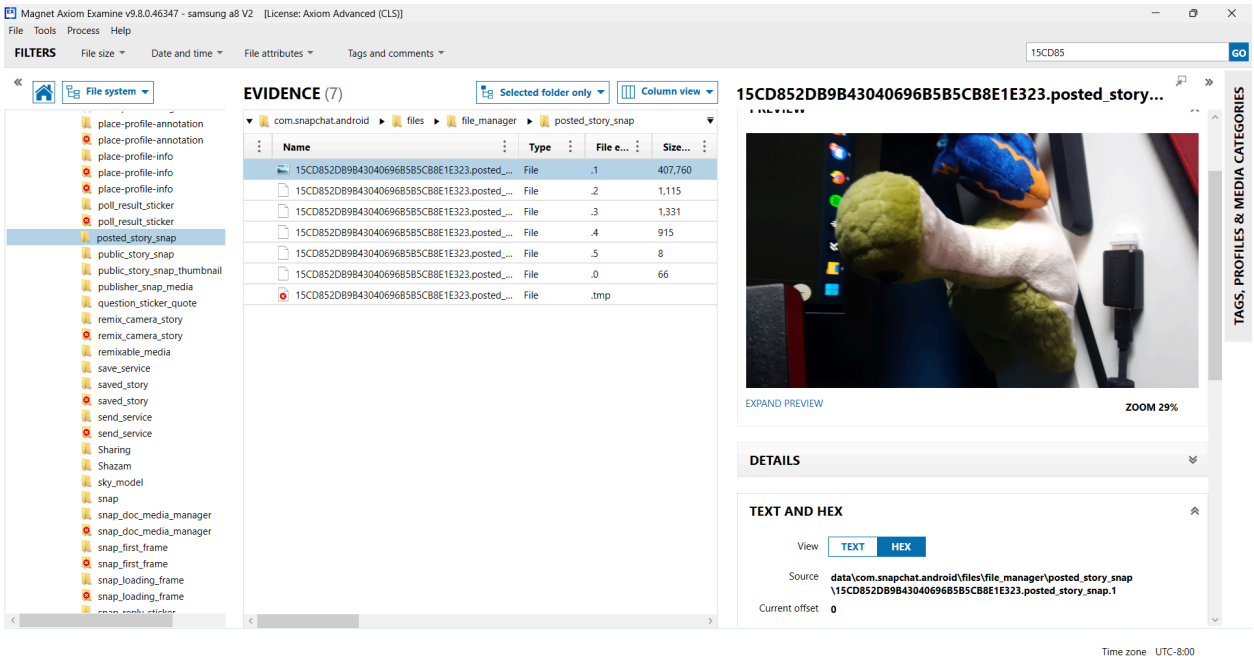


Figure F2. Posted story snap media recovered.

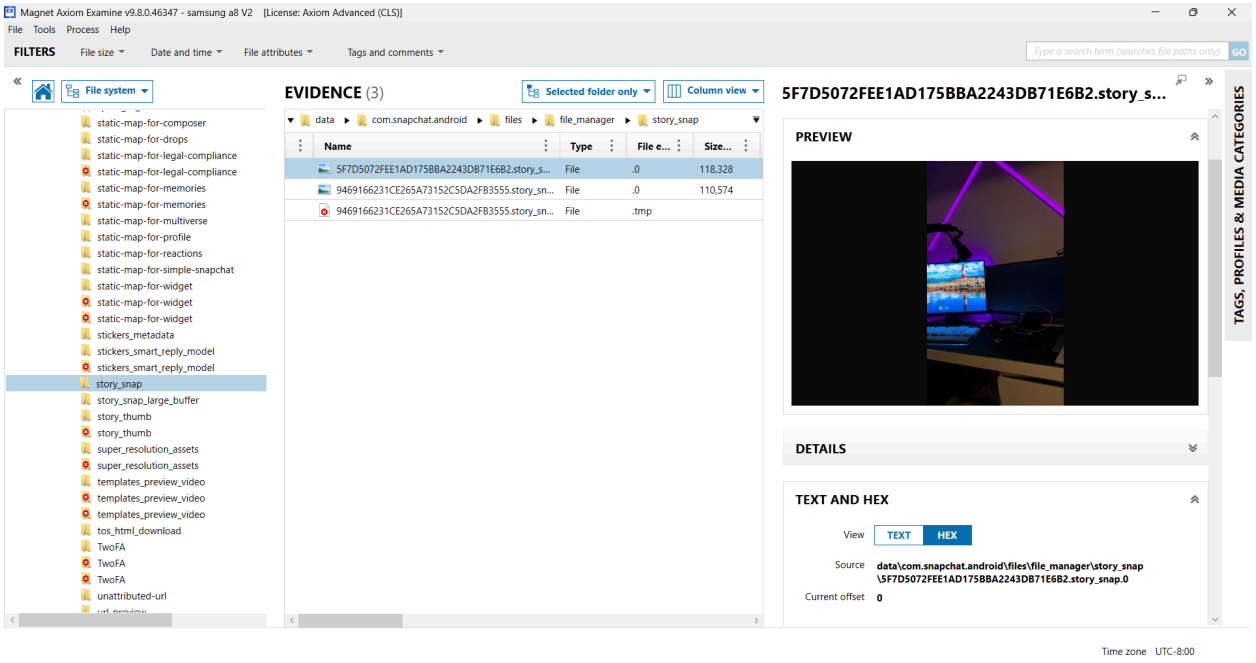


Figure F3. Viewed Snapchat story recovered in story_snap.

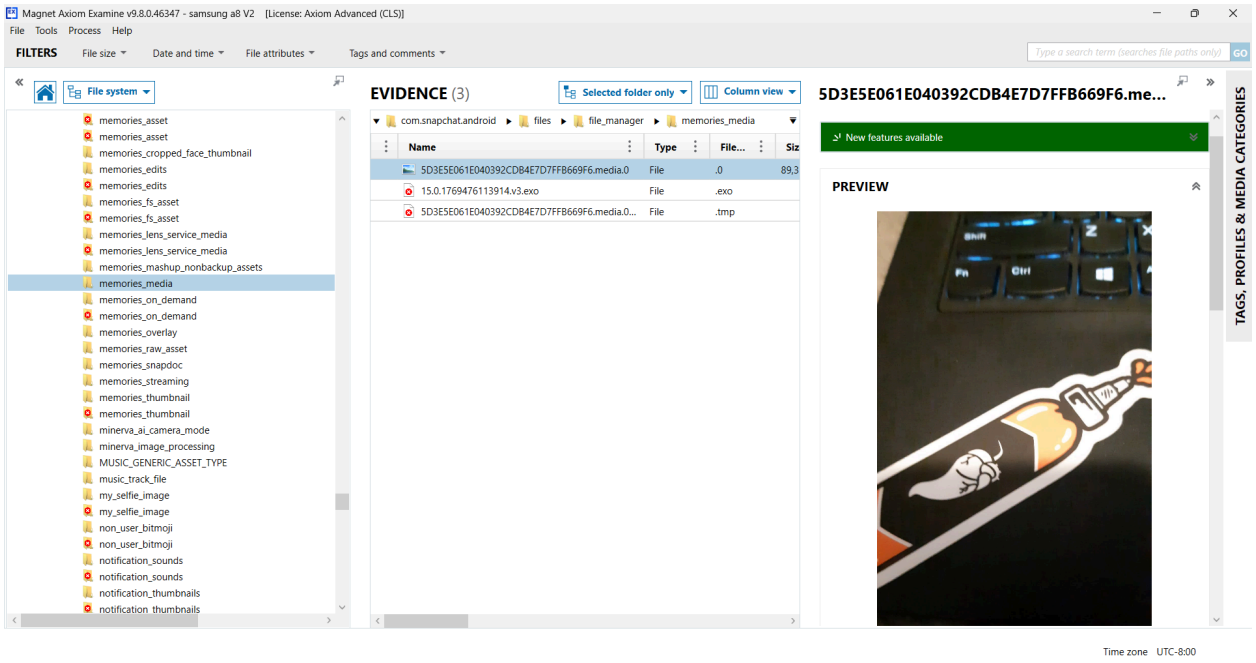


Figure F4. Saved memory cache recovered in memories_media.

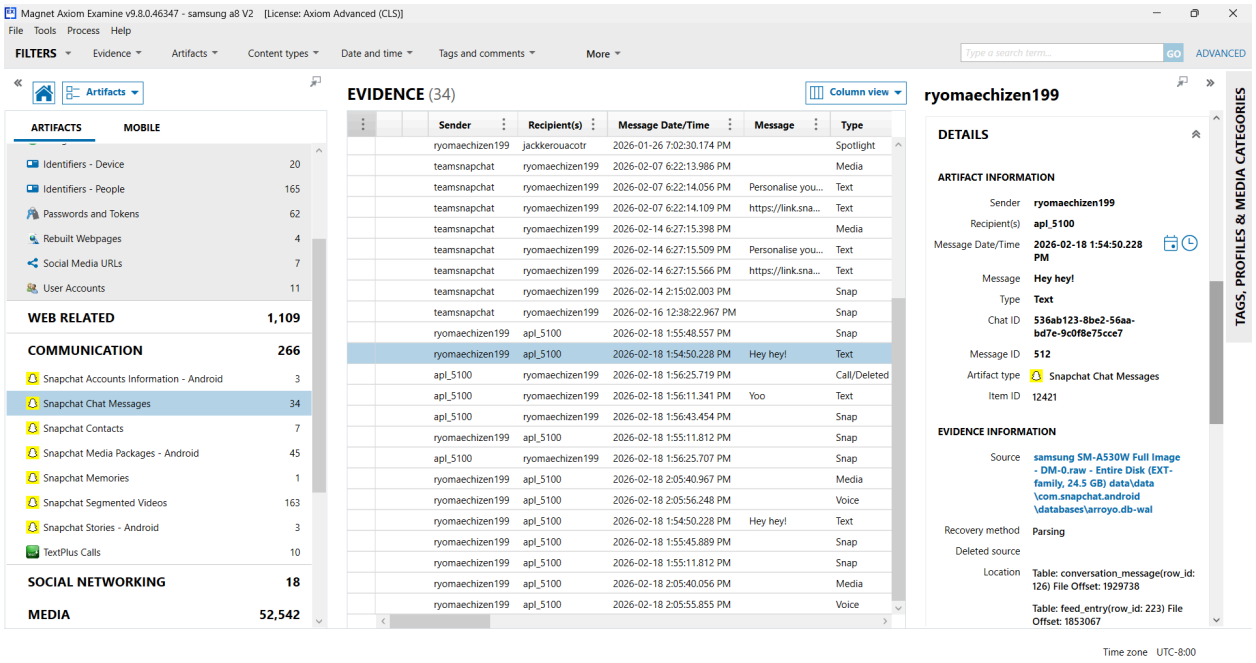


Figure F5. Chat text recovered in the Snapchat category.

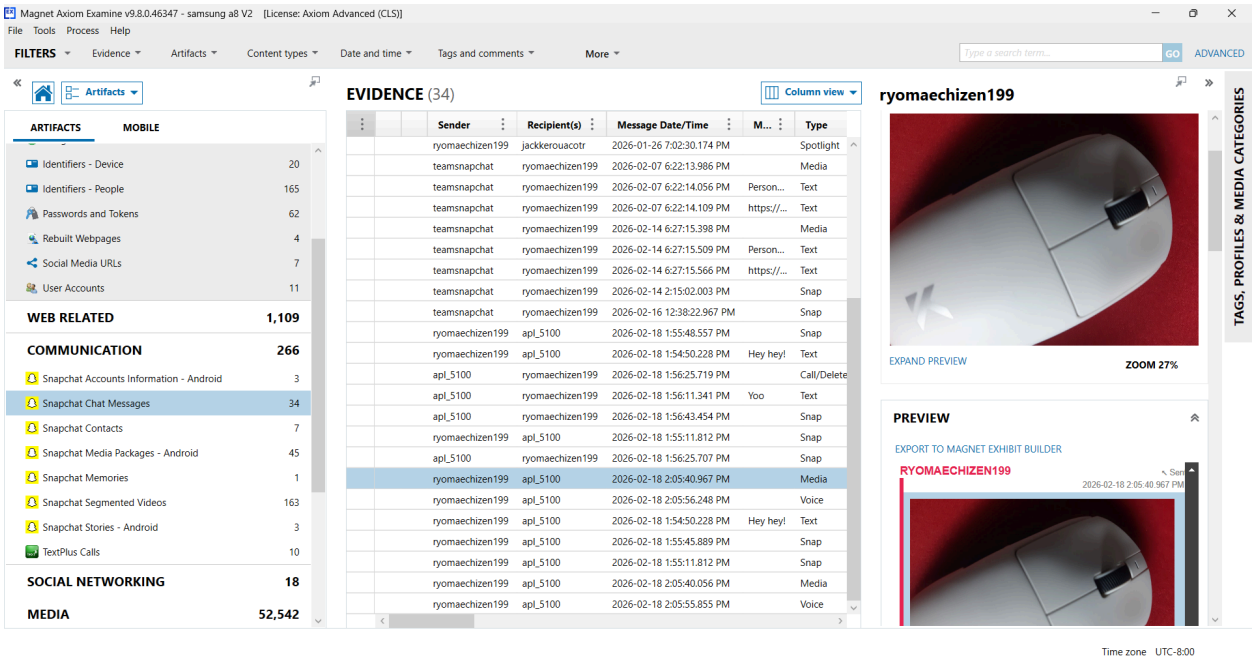


Figure F6. Recovered sent photo in chat in Snapchat category.

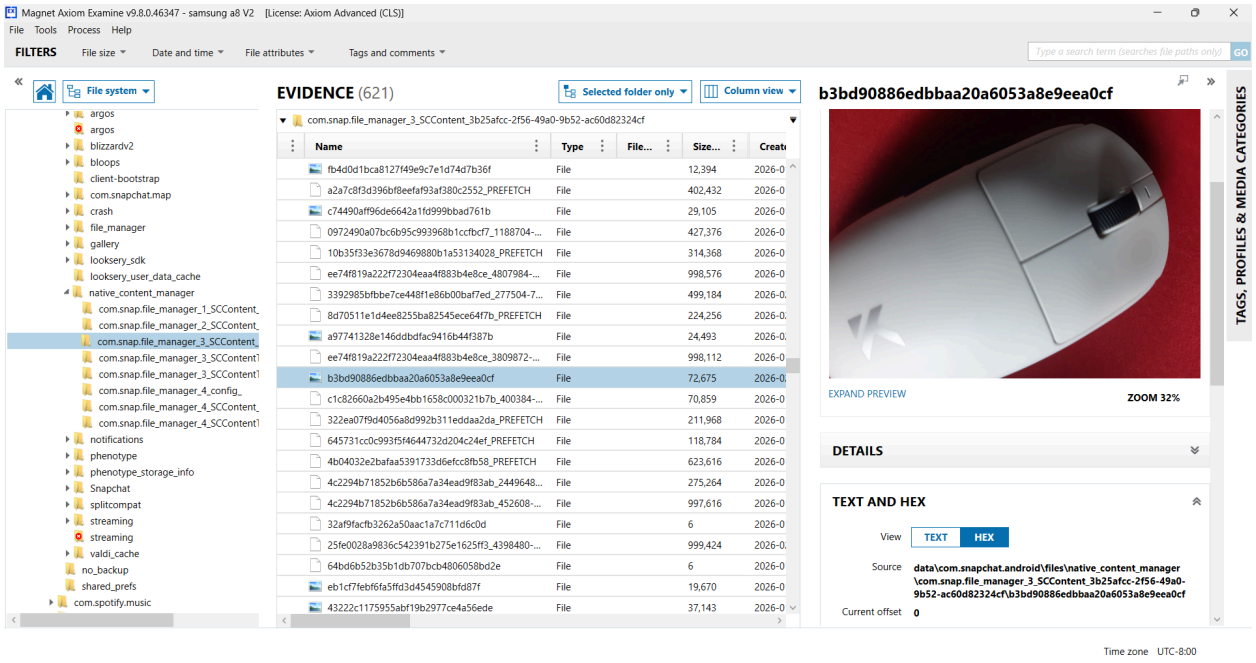


Figure F7. Sent photo cache recovered in native_content_manager.

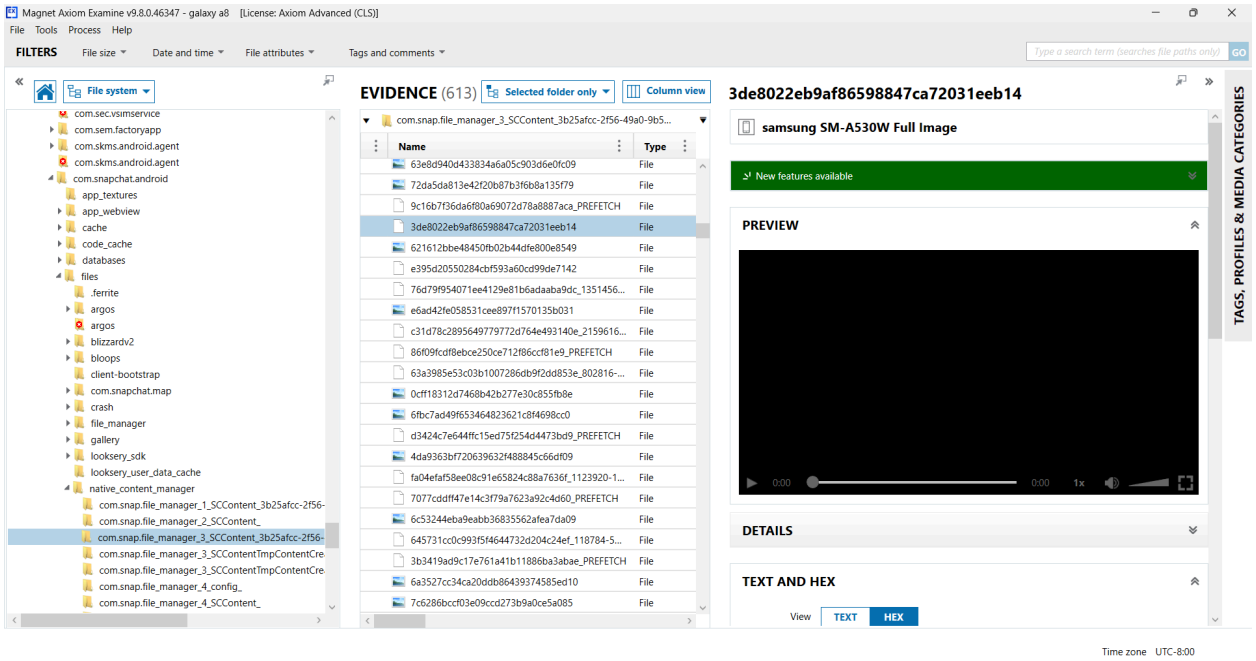


Figure F8. Recovered voice note cache in native_content_manager.

Appendix G: Samsung Galaxy S8 Artifact Screenshots

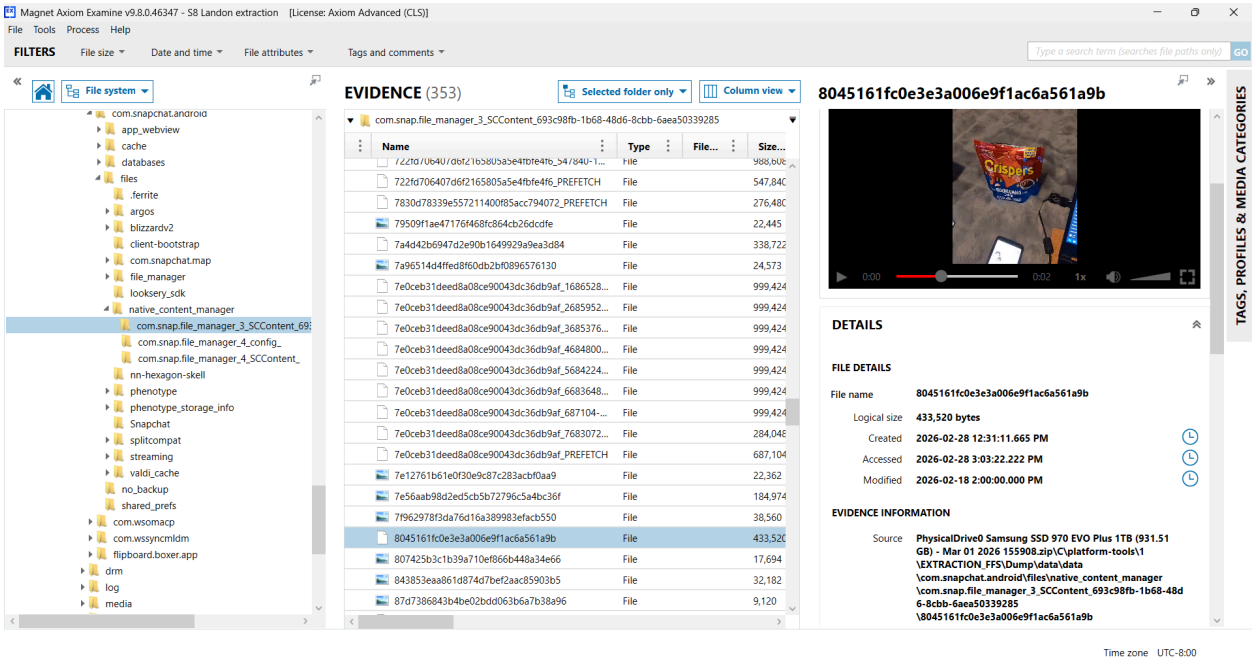


Figure G1. Recovered video snap media cache in native_content_manager.

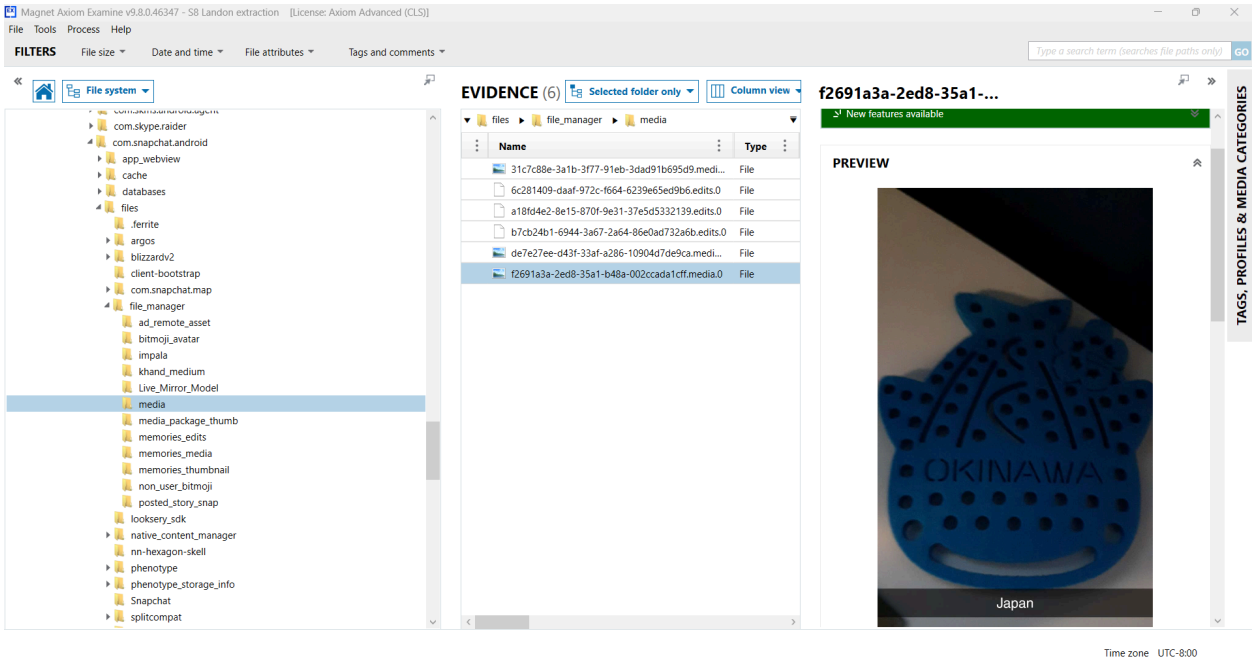


Figure G2. Recovered received snap media cache in media.

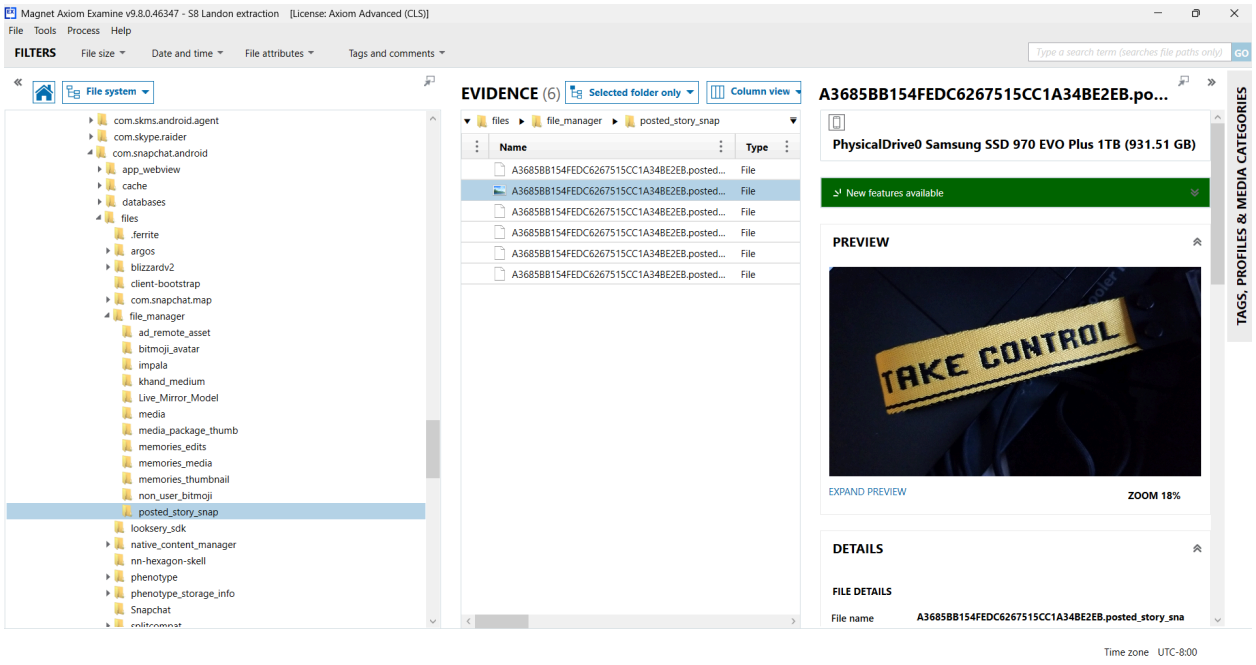


Figure G3. Recovered snap story media cache in posted_story_snap.

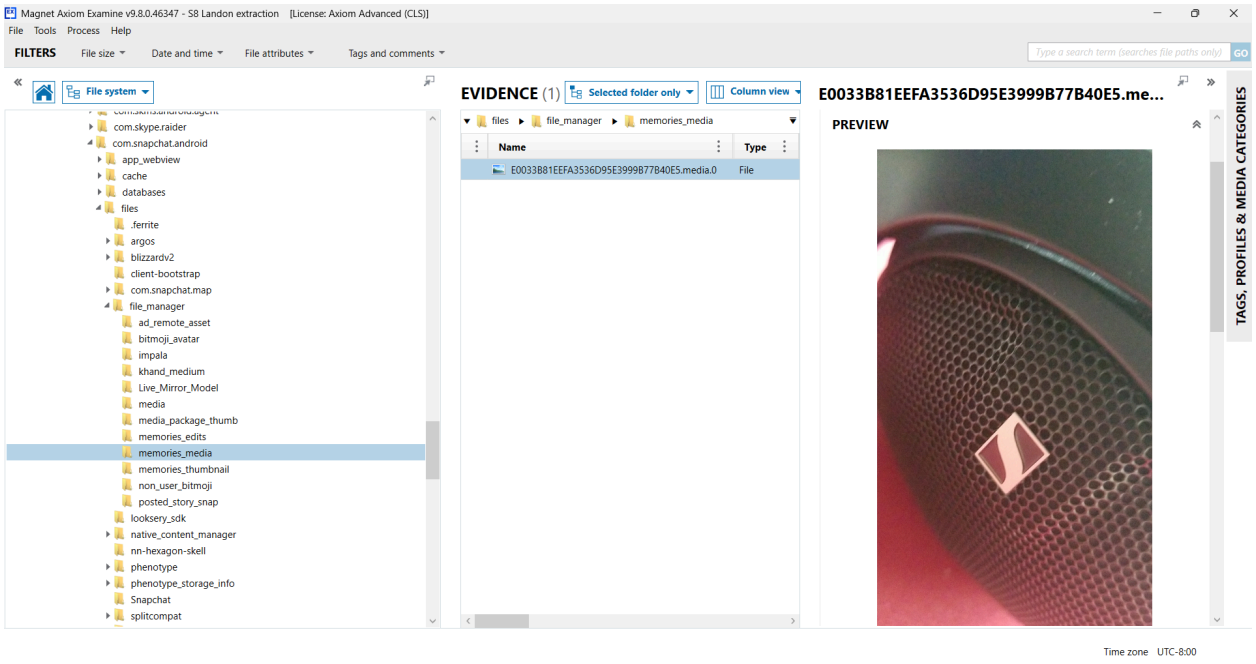


Figure G4. Recovered saved memory media cache in memories_media.

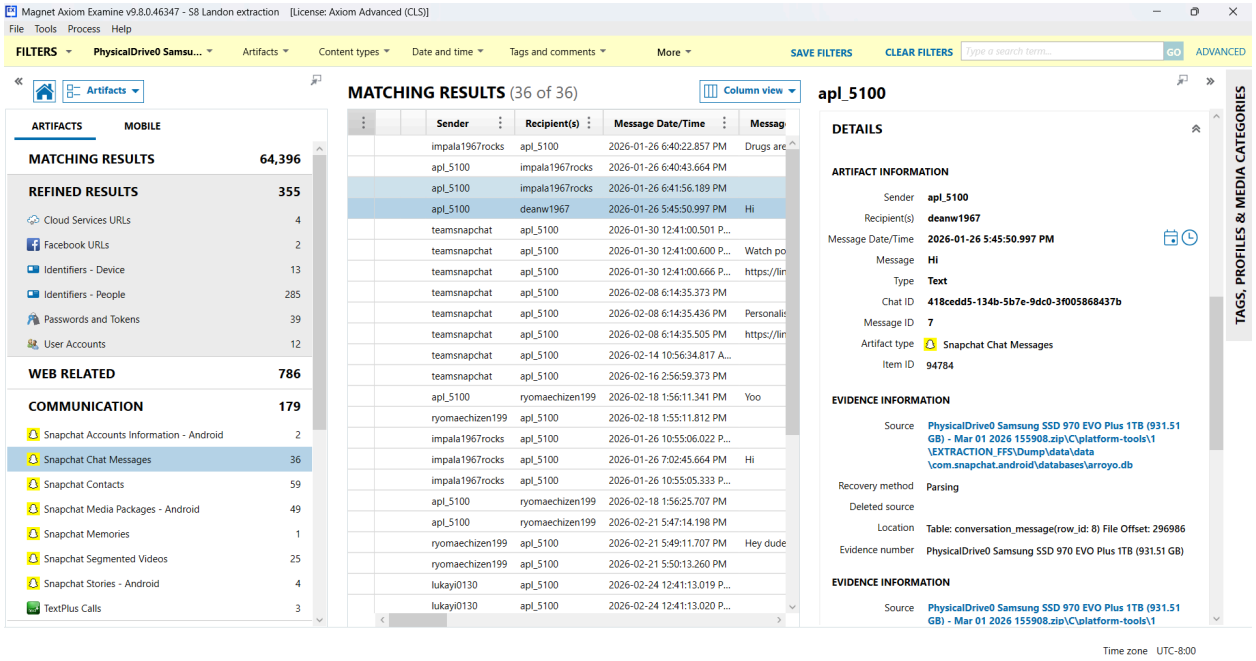


Figure G5. Recovered chat text message artifact in Snapchat Chat Messages Category.

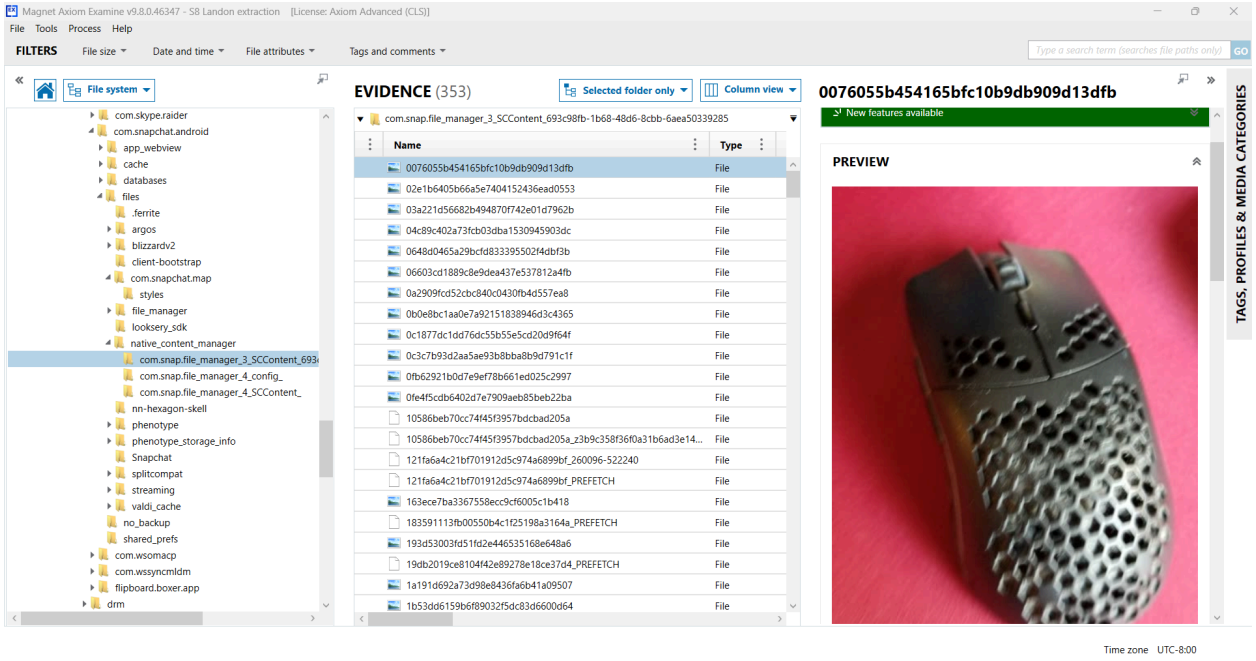


Figure G6. Recovered sent photo media cache in native_content_manger.

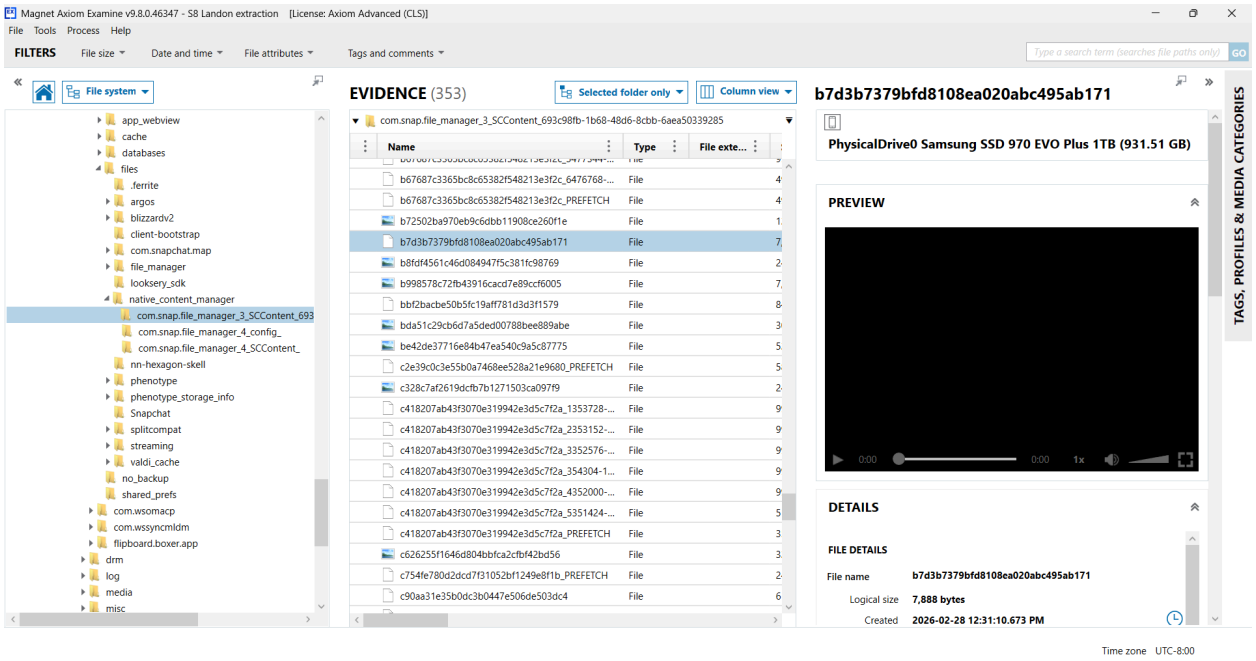


Figure G7. Recovered voice note media cache in native_content_manager.

Appendix H: Additional iOS Device Artifacts

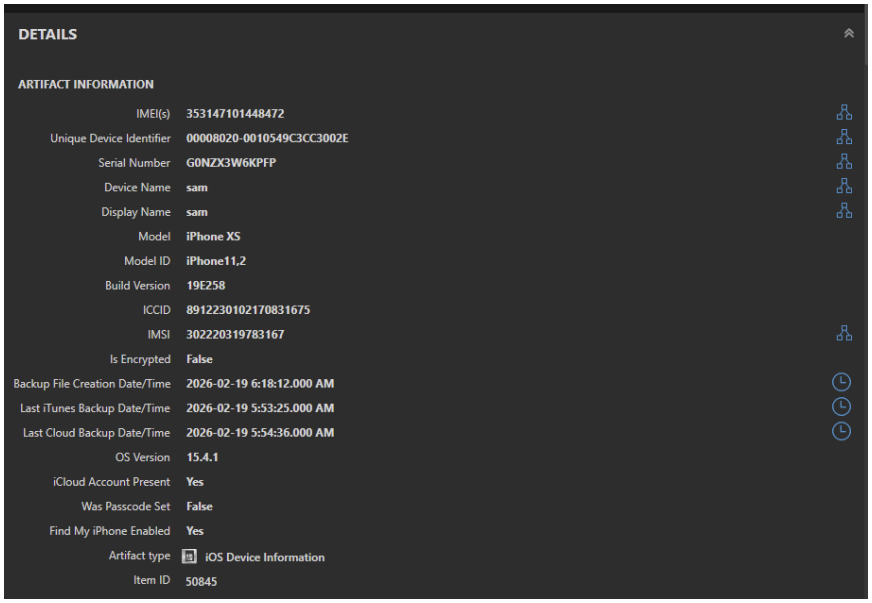


Figure H1. Axiom artifact showing Sam Winchester's device identification details

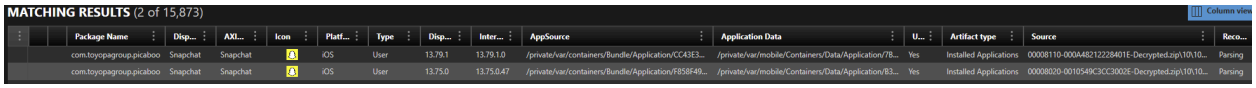


Figure H2. Installed Snapchat application artifacts in Axiom

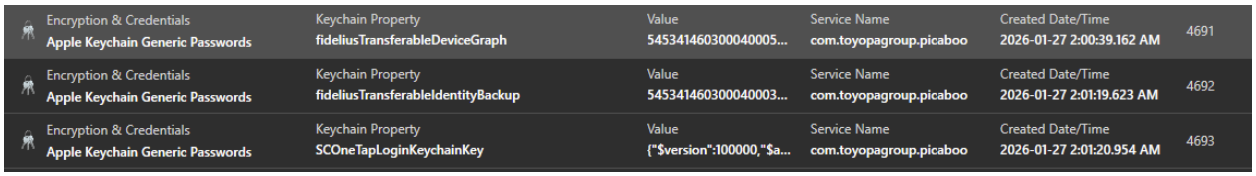


Figure H3. Axiom keychain artifacts for Sam Winchester's Snapchat account on the iPhone XS (Items 4691-4693)

```
func (a *Account) InitializeWebKey() (*types.FideliusKeys, error) {
    headers := headers.NewCoreHeaders(a.client.Session.SnapCookies, a.client.Session.SnapTokens, a.client.device)
    tentativeWebKey, pubKey, privKey, err := crypto.NewFideliusTentativeWebKey()
```

(Oxzer, 2023) Figure H4. Github source showing the InitializeWebKey endpoint for Fidelius key generation

Item ID	Artifact	Supporting detail	Date and time
6986	Application Usage **InteractionC Interactions**	Display Name **Sam W**	Created Date/Time **2026-02-13 3:00:53.729 AM**
6987	Application Usage **InteractionC Interactions**	Display Name **Sam W**	Created Date/Time **2026-02-13 3:08:09.436 AM**
6990	Application Usage **InteractionC Interactions**		Created Date/Time **2026-02-13 3:09:30.001 AM**
6992	Application Usage **InteractionC Interactions**	Display Name **Sam W**	Created Date/Time **2026-02-13 3:09:42.781 AM**
6994	Application Usage **InteractionC Interactions**	Display Name **Sam W**	Created Date/Time **2026-02-13 3:12:44.354 AM**
6995	Application Usage **InteractionC Interactions**	Display Name **Sam W**	Created Date/Time **2026-02-13 3:12:54.656 AM**
6996	Application Usage **InteractionC Interactions**		Created Date/Time **2026-02-13 3:13:09.023 AM**
6997	Application Usage **InteractionC Interactions**	Display Name **Sam W**	Created Date/Time **2026-02-13 3:14:09.127 AM**
6998	Application Usage **InteractionC Interactions**		Created Date/Time **2026-02-13 3:15:00.880 AM**
6999	Application Usage **InteractionC Interactions**		Created Date/Time **2026-02-13 3:15:28.291 AM**
7000	Application Usage **InteractionC Interactions**		Created Date/Time **2026-02-13 3:16:44.965 AM**
7001	Application Usage **InteractionC Interactions**		Created Date/Time **2026-02-13 3:17:49.527 AM**
7002	Application Usage **InteractionC Interactions**	Display Name **Sam W**	Created Date/Time **2026-02-13 3:18:30.827 AM**
7003	Application Usage **InteractionC Interactions**	Display Name **Sam W**	Created Date/Time **2026-02-13 3:18:39.049 AM**
7004	Application Usage **InteractionC Interactions**	Display Name **Sam W**	Created Date/Time **2026-02-13 3:18:47.444 AM**
7005	Application Usage **InteractionC Interactions**		Created Date/Time **2026-02-13 3:19:13.323 AM**

Figure H5. Longest recorded conversation identified through InteractionC events

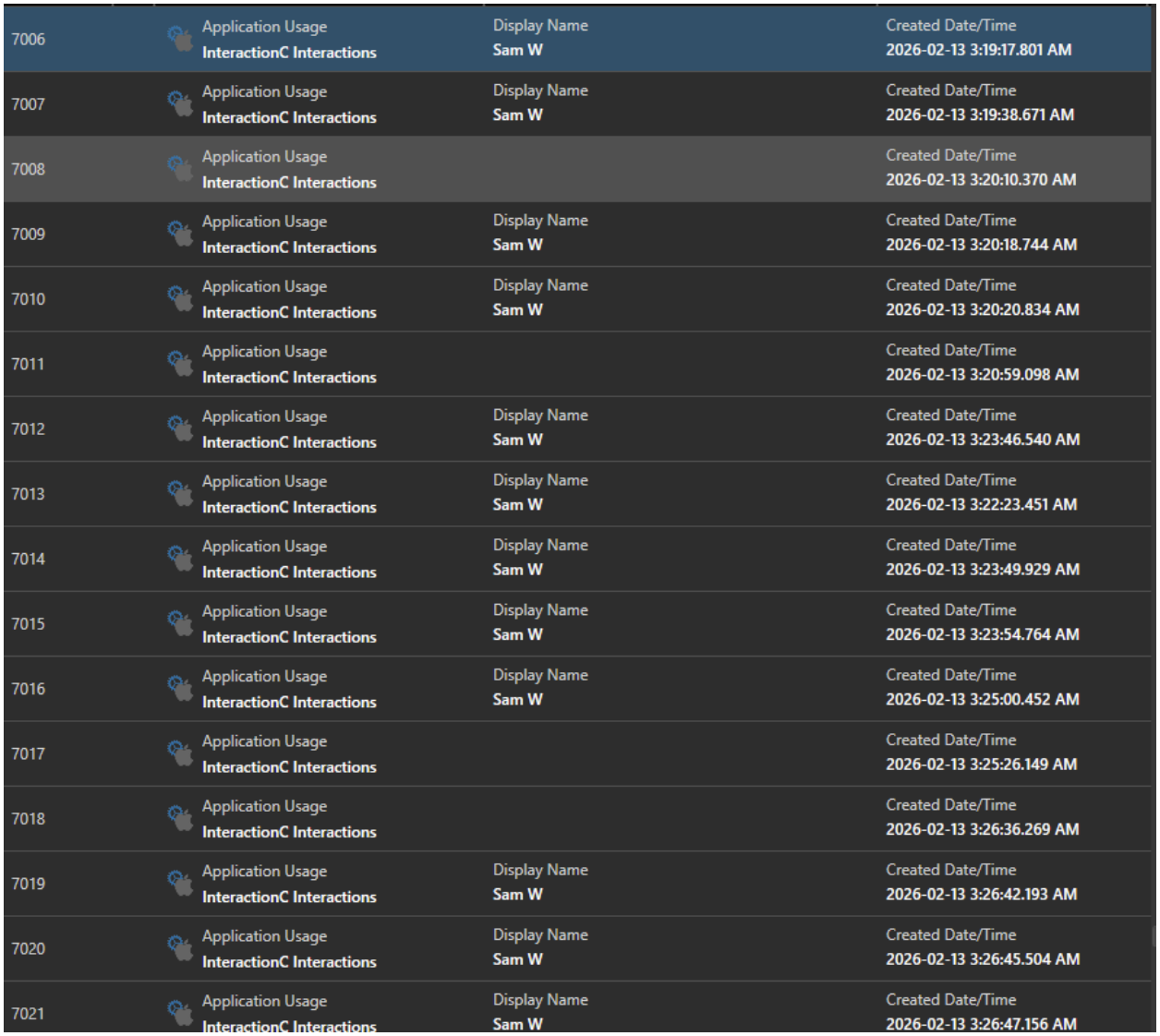


Figure H6. Longest recorded conversation identified through InteractionC events.

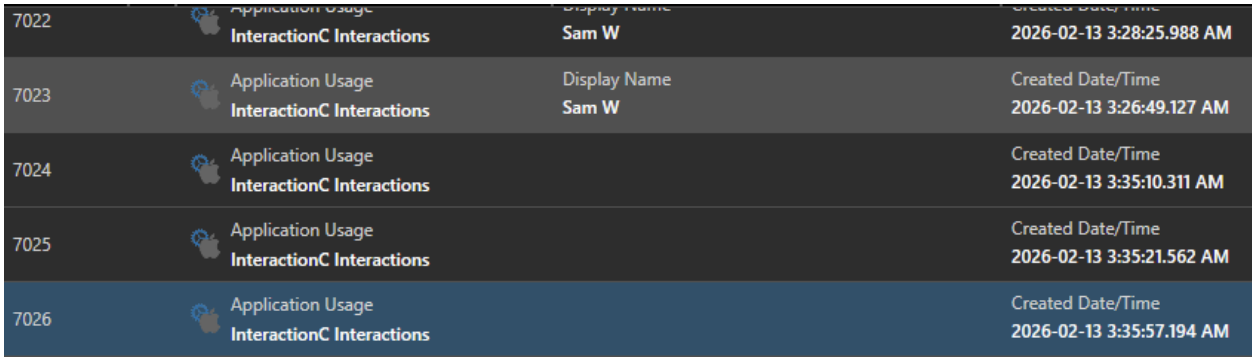


Figure H7. Longest recorded conversation identified through InteractionC events.

7027	Application Usage InteractionC Interactions	Display Name Sam W	Created Date/Time 2026-02-19 6:10:55.065 AM
7028	Application Usage InteractionC Interactions	Display Name Sam W	Created Date/Time 2026-02-19 6:11:00.776 AM
7029	Application Usage InteractionC Interactions		Created Date/Time 2026-02-19 6:11:08.752 AM
7030	Application Usage InteractionC Interactions		Created Date/Time 2026-02-19 6:11:26.166 AM
7031	Application Usage InteractionC Interactions	Display Name Sam W	Created Date/Time 2026-02-19 6:11:36.205 AM
7032	Application Usage InteractionC Interactions	Display Name Sam W	Created Date/Time 2026-02-19 6:11:39.418 AM
7033	Application Usage InteractionC Interactions		Created Date/Time 2026-02-19 6:11:46.566 AM
7034	Application Usage InteractionC Interactions	Display Name Sam W	Created Date/Time 2026-02-19 6:12:06.084 AM
7035	Application Usage InteractionC Interactions	Display Name Sam W	Created Date/Time 2026-02-19 6:13:10.788 AM
7036	Application Usage InteractionC Interactions		Created Date/Time 2026-02-19 6:13:47.113 AM
7037	Application Usage InteractionC Interactions		Created Date/Time 2026-02-19 6:14:04.301 AM

Figure H8. Last recorded conversation identified through InteractionC events.

6949	Application Usage InteractionC Interactions	Display Name Video for Dean	Created Date/Time 2026-01-28 1:58:57.821 AM
6950	Application Usage InteractionC Interactions	Display Name Dean, Try This Lens!	Created Date/Time 2026-01-28 5:03:01.199 AM
6951	Application Usage InteractionC Interactions	Display Name Video for Dean	Created Date/Time 2026-01-28 5:31:14.320 PM
6954	Application Usage InteractionC Interactions	Display Name The Rundown	Created Date/Time 2026-01-28 8:33:20.382 PM
6957	Application Usage InteractionC Interactions	Display Name Team Snapchat	Created Date/Time 2026-01-28 11:43:23.222 PM
6960	Application Usage InteractionC Interactions	Display Name Video for Dean	Created Date/Time 2026-01-30 2:38:36.493 AM
6962	Application Usage InteractionC Interactions	Display Name Dean, Try This Lens!	Created Date/Time 2026-01-30 5:42:42.393 AM
6964	Application Usage InteractionC Interactions	Display Name Video for Dean	Created Date/Time 2026-01-30 3:01:47.529 PM
6965	Application Usage InteractionC Interactions	Display Name NFL Audible	Created Date/Time 2026-01-30 6:06:55.268 PM
6966	Application Usage InteractionC Interactions	Display Name Video for Dean	Created Date/Time 2026-01-31 3:20:06.793 AM

ARTIFACT INFORMATION

Bundle ID	com.toyopagroup.picaboo
Display Name	Video for Dean
Sender	Video for Dean
Created Date/Time	2026-01-28 1:58:57.821 AM
Start Date/Time	2026-01-28 1:58:57.669 AM
End Date/Time	2026-01-28 1:58:57.669 AM
Artifact type	InteractionC Interactions
Item ID	6949

EVIDENCE INFORMATION

Source	00008110-000A48212228401E-Decrypted.zip\1f\1f5a521220a3ad80ebfdc196978df8e7a2e49dee
Recovery method	Parsing
Deleted source	
Location	Table: ZINTERACTIONS(Z_PK: 19) Table: ZCONTACTS(Z_PK: 3)
Evidence number	PhysicalDrive2 Samsung SSD 850 EVO 2TB (1.82 TB) - C\Users\Kaiden\Desktop\00008110-000A48212228401E-Decrypted

Figure H9. Notifications for personalized suggestions and pages in InteractionC events

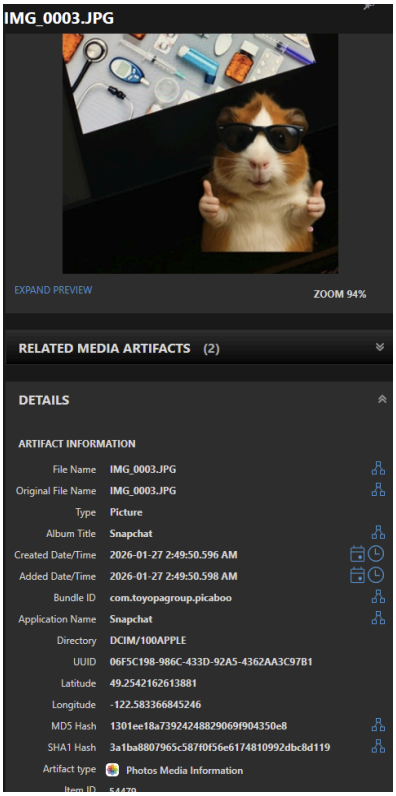


Figure H10. AXIOM Examine identifying saved photo as Snapchat-originated item, with EXIF data showing the precise capture location.

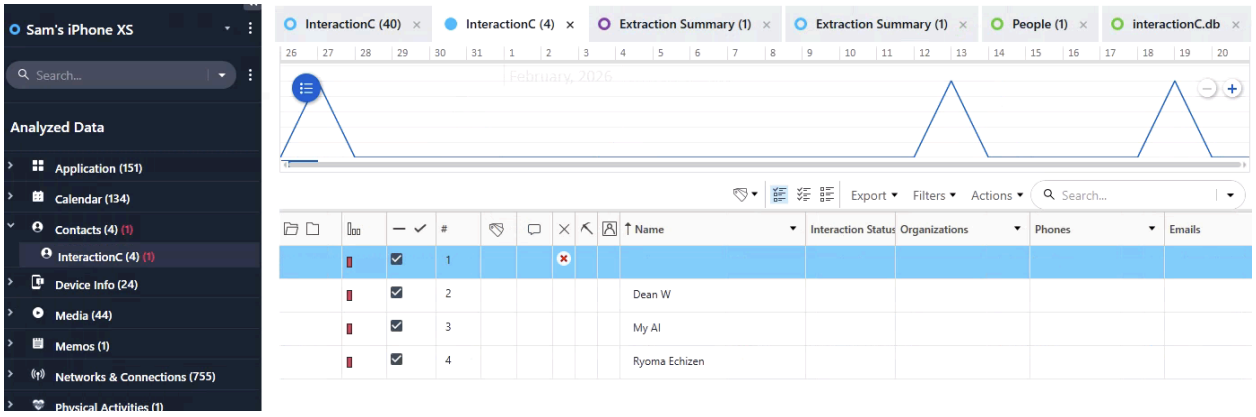


Figure H11. Cellebrite Inseyets dedicated contact section

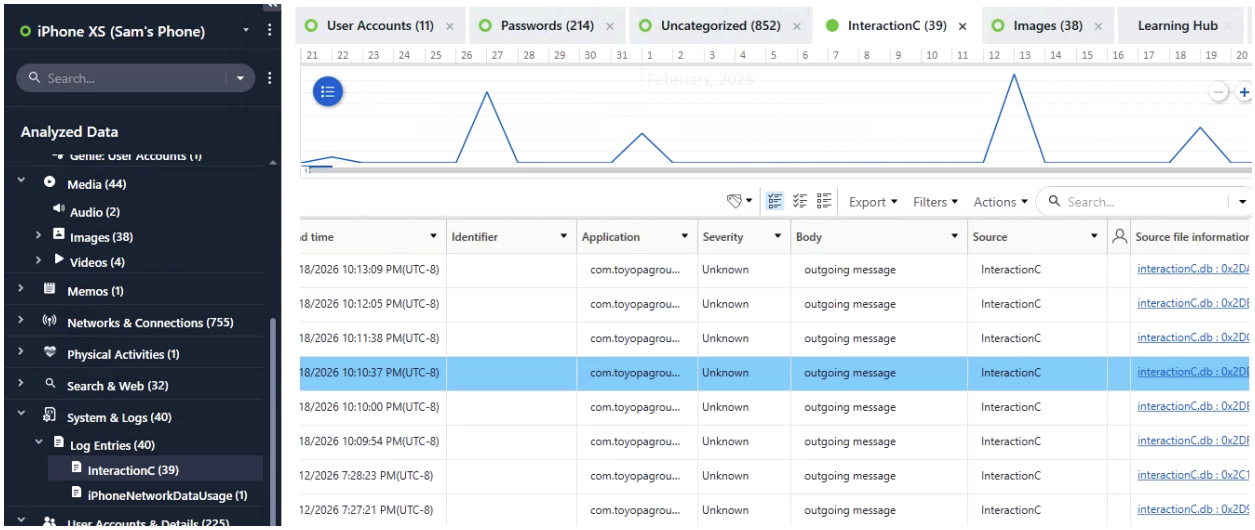


Figure H12. Cellebrite Inseyets InteractionC event view without display names