



Prifysgol Cymru
Y Drindod Dewi Sant
University of Wales
Trinity Saint David

ACFA7003 - Master's Project

Student Name:
SMIT BRIJESHKUMAR PATEL

Student ID:
2506590

Programme:
MSc Cybersecurity and Digital Forensics
Research Topic:
Evaluating Cellebrite UFED and UFED Physical Analyzer Against
WhatsApp, Signal and Telegram on Android

Submission Date:
Friday, 25th September 2026

Table of Contents

Chapter 1: Introduction.....	4
1.1 Background.....	4
1.2 Problem Statement.....	5
1.3 Aim and Objectives	5
1.4 Research Questions.....	6
1.5 Scope and Delimitations	6
1.6 Significance of the Study	6
1.7 Dissertation Structure	6
1.8 Traceability Matrix	7
Chapter 2: Literature Review.....	7
2.1 Mobile Forensics Fundamentals	7
2.2 Cellebrite UFED and UFED Physical Analyzer	8
2.3 Encrypted Messaging Apps	9
2.4 Prior Tool-Evaluation Studies	10
2.5 Legal and Evidentiary Context	11
2.6 Identified Gap	12
Chapter 3: Methodology	12
3.1 Research Philosophy and Approach	12
3.2 Encryption Architecture Analysis (O1)	13
3.3 Experimental Design and Dataset.....	15
3.4 Data Collection	15
3.5 Evaluation Criteria.....	16
3.6 Ethics	17
Chapter 4: Results and Troubleshooting.....	17
4.1 Acquisition Log	17
4.2 File System (Android Backup) Attempts.....	19
4.3 Advanced Logical Attempts	20
4.4 Troubleshooting Applied	20
4.5 Decoding and Recovery Scoring	21
4.6 Interpretation	22
Chapter 5: Legal and Evidential Implications	23
5.1 Admissibility	23
5.2 Chain of Custody	24
5.3 Jurisdictional Considerations	24
5.4 Ethical Issues	25
Chapter 6: Discussion and Practitioner Framework	25
6.1 Synthesis.....	25
6.2 Proposed Framework.....	26

6.3 Discussion against the Literature	27
Chapter 7: Conclusion	28
7.1 Summary per Objective	28
7.2 Contribution.....	29
7.3 Limitations	29
7.4 Future Work.....	30
References	31
Appendices	34
Appendix A: Extraction logs and screenshots	34
Appendix B: Hash values and chain-of-custody records	42
Appendix C: Device and app configuration details	44
Appendix D: Ethics approval documentation	50

Evaluating Cellebrite UFED and UFED Physical Analyzer Against WhatsApp, Signal and Telegram on Android

Chapter 1: Introduction

Message content in WhatsApp, Signal and Telegram is encrypted end-to-end by default. For a forensic examiner this narrows the useful line of enquiry from interception to whatever survives in readable form on the handset itself. This chapter sets out that problem and the response the dissertation makes to it. Section 1.1 describes how the three applications became routine in casework, and how Android's own encryption has reshaped what acquisition can achieve. Section 1.2 identifies the gap the study addresses: vendors publish detailed capability claims, but independent testing of those claims against these applications is limited, and much of what exists is now dated. Sections 1.3 and 1.4 give the aim, the four objectives and the research questions. Section 1.5 sets the boundaries of the work, Section 1.6 explains who the findings are for, and Sections 1.7 and 1.8 set out how the remaining chapters are arranged and how each question is traced through them.

1.1 Background

Instant messaging has displaced SMS as the primary channel for personal communication, and WhatsApp has the largest user base of the three applications examined here [8]. Signal and Telegram are smaller but have grown on explicit privacy claims. The consequence for criminal investigation is that communications which once existed in a network operator's records now sit inside applications the operator cannot read.

Onik et al. [7] attribute the difficulty this creates to the widespread adoption of end-to-end encryption, which protects message content and multimedia at rest as well as in transit and leaves examiners dependent on the artefacts an application leaves behind. Interception, in other words, has stopped being the useful route to message content, and what remains is the question of how much survives in recoverable form on the handset.

Android is where that question arises most often, partly because of its market position and partly because the diversity of its hardware and vendor implementations complicates the choice of forensic tool [1]. The platform itself has also shifted underneath examiners. Full Disk Encryption arrived at Android 5.0 and File-Based Encryption became mandatory for devices launching with Android 10 [20], with the effect that a bit-for-bit image of the user data partition now yields ciphertext. Physical extraction has largely ceased to be viable, and practice has moved towards full file system extraction from a device that is running and unlocked.

Examiners therefore rely on commercial forensic platforms to close the distance between an encrypted application store and readable evidence. Fakiha [2] treats that reliance as a defining constraint on the discipline, alongside device heterogeneity and the fragmentation of data between local and cloud storage. The reliance is not itself a problem. It becomes one when a platform fails without indicating that it has failed, since the examiner has no independent means of knowing what was not recovered.

1.2 Problem Statement

Forensic examiners depend on commercial extraction platforms, and most of what is known about their capability originates with the companies that sell them. Cellebrite publishes supported-application documentation for UFED and Physical Analyzer. Whether that documentation survives contact with examination conditions is an empirical question rather than a documentary one.

Independent testing of the platform comes from one place, the NIST Computer Forensics Tool Testing programme, and each published cycle records different shortfalls from the one before [5], [6]. CFTT checks categories of data, not how well a tool decodes a particular application. WhatsApp does appear, under social media, but the latest cycle only records whether account details came back, and on one handset they came back only in part [6]. Signal and Telegram are not tested. So the most authoritative check on this tool has almost nothing to say about the question asked here.

Where decoding has been tested rather than assumed, the variation is wide enough to affect case outcomes. Riadi et al. [10] scored three commercial platforms against Signal on Android across eleven parameters and recorded 78.69, 26.23 and 9.84 percent accuracy on identical data. Cellebrite was not among the three. Vasilaras et al. [14] found UFED Physical Analyzer 7.50 decoding Telegram normal chat media but not secret chat media, and Magnet AXIOM 6.0 decoding media that version 5.7 could not, so what the evidence appears to contain depends on the platform and on its version number.

For the Cellebrite platform across all three applications the question is still open, and the closest studies each stop short in a different direction. Sinaga et al. [11] compare UFED against its rivals but stay with logical extraction and report nothing at application level for encrypted messaging. Heath et al. [3] use Cellebrite against WhatsApp and Telegram, again only under logical and advanced logical extraction, so the acquisition method now standard on Android goes untested. Riadi et al. [10] test Signal in depth without testing UFED.

1.3 Aim and Objectives

This study evaluates how much of WhatsApp, Signal and Telegram Cellebrite UFED and Physical Analyzer recover from an Android 16 handset, and what follows where a tool performs below its documented capability. Four objectives support that aim.

- O1. Establish what has been tested across mobile acquisition, the Cellebrite platform and encrypted messaging artefacts, what remains untested, and how each application encrypts and stores its data, so that Chapter 4 can explain its results rather than only report them.
- O2. Design and execute a controlled extraction across all three applications on identical seeded data under an established forensic methodology.
- O3. Measure tool performance for completeness, artefact type, accuracy and acquisition time, and compare the three applications on the first three.
- O4. Assess the admissibility and chain-of-custody consequences of the results under Directive (EU) 2016/680, then produce a practitioner framework setting out which acquisition methods to attempt and what must be recorded when a tool returns nothing.

1.4 Research Questions

Three questions follow from the aim, one for each stage of the work.

RQ1. How do WhatsApp, Signal and Telegram store and protect message and media data on Android 16, and what does existing testing establish about the Cellebrite platform's capability against those structures?

RQ2. How much of a known seeded corpus do UFED and Physical Analyzer recover from each application under the acquisition methods available on an unrooted Android 16 handset, where do the three applications differ in completeness, artefact type and accuracy, and how long does each method take?

RQ3. What are the admissibility and chain-of-custody consequences under Directive (EU) 2016/680 where an acquisition method fails or a tool returns no data, and what must an examiner record in that situation?

1.5 Scope and Delimitations

The study covers Android 16 on physical handsets. iOS, desktop and web clients are excluded, because they present different acquisition problems and treating them together would produce a weaker answer to each. Application versions are recorded before extraction, since one minor release can change what a tool decodes [14]. Cellebrite is the only platform tested, so this describes one toolset rather than ranking it against XRY or AXIOM [11]. The test corpus is seeded rather than real. That resolves the consent problem and creates another, because seeded conversations are tidier than real ones, even on a handset in daily use. Cloud content and network interception fall outside the scope; only artefacts resident on the device at seizure are examined [2]. The legal frame is European Union law, principally Directive (EU) 2016/680 [16], so the admissibility conclusions should not be read as holding elsewhere.

1.6 Significance of the Study

Every artefact on the test device was placed there by the researcher, so a gap in the output is a measured failure rather than an open question. That is the difference between reporting that a tool found nothing and reporting that a tool failed to recover a known message. An examiner can name the artefacts each application is expected to yield and log the rest as tested absences or as not acquired. An expert witness can attribute a limitation to independent testing rather than to vendor documentation [6]. A later study gains an Android 16 baseline with the build and patch level recorded at each session. The closest prior work lacks one, since it runs on earlier releases or omits one of the three applications [3], [11]. Forensic method assurance depends on validation obtained this way [5], [15].

1.7 Dissertation Structure

Chapter 2 reviews the literature and defines the gap. Chapter 3 sets out the method, including the encryption analysis that later interpretation depends on. Chapter 4 reports the acquisition attempts, the troubleshooting applied and the recovery results, and explains where the acquisition stopped. Chapter 5 examines whether those results hold up legally and ethically. Chapter 6 develops a practitioner framework and discusses the findings against existing work. Chapter 7 reports the outcome against each objective, the contribution made and the limitations.

1.8 Traceability Matrix

RQ	Objective	Chapter and section	Evaluation metric
RQ1	O1	Ch 2 (2.1–2.6); 3.2	Storage location, storage form, protection mechanism and key custody documented per application; one stated recovery expectation per artefact class
RQ2	O2, O3	3.3–3.5; Ch 4 (4.1–4.6)	Recovery state per item against the ground truth log (fully recovered, partially recovered, present but undecoded, absent, not acquired); completeness as a percentage per application per method; SHA-256 match rate for media; acquisition time per method; method outcome recorded as completed, partially completed or failed
RQ3	O4	Ch 5; Ch 6 (6.2)	Doctrinal assessment against Articles 4, 6 and 25 of Directive (EU) 2016/680; framework evaluated on coverage of the acquisition ladder and of the fields to be logged where a method fails or returns nothing

Chapter 2: Literature Review

The literature that matters here was written in three places that do not cite one another often. Android acquisition research establishes what platform encryption leaves available to any tool [1],[15]. Cellebrite evaluation, chiefly the NIST test cycles, measures the platform against generic data categories and gets on closer to individual applications than WhatsApp account details [5],[6]. Application-level forensics goes deep on WhatsApp, Signal or Telegram, but takes one at a time and rarely uses Cellebrite to do it [8],[10],[14]. This chapter reads them against one another. Section 2.1 establishes the acquisition types and custody requirements the later chapters rely on, and Section 2.2 sets Cellebrite's documented capability beside what independent testing found. Section 2.3 takes each application in turn, since storage design decides what a tool can decode. Section 2.4 examines how prior evaluations were run, an issue that determines how far their findings transfer. Sections 2.5 and 2.6 cover the evidentiary requirements and the gap this study addresses.

2.1 Mobile Forensics Fundamentals

Acquisition methods are ranked by how deep into the storage they reach. Manual examination captures what the screen shows and nothing else. Logical acquisition asks the operating system or a backup service for data and gets back what the device is willing to hand over. File system acquisition takes the directory structure, which brings in the databases and journal files a logical request leaves behind. Physical acquisition images the medium bit for bit. NIST SP 800-101r1 defines these levels and the procedure attached to each, and Android studies still work to it [13],[17].

Physical acquisition used to be the method that recovered deleted content, because unallocated space came with the image. File-Based Encryption ended that for handsets launching with Android 10 onward, since the image is ciphertext and the keys sit with the running device [15]. Full file system acquisition from a live unlocked phone is now the standard route on Android, and the examiner's dependency has moved with it, off the storage medium and onto whether the tool can obtain and apply the keys.

That move costs something evidentially. Integrity has conventionally been demonstrated by hashing a static image at acquisition and reproducing the value later. Yoo et al. [15] found the model breaks on full file system extractions. The device keeps writing while the extraction runs, so two acquisitions of the same handset return different hashes. Integrity then has to be shown file by file, and the examiner has to be able to account for why the two extractions differ.

Chain of custody is the continuous record of who held the device, when, and what was done to it, running from seizure through to the report. Hash verification is its digital half, and it is what establishes that the extraction the report relies on is the one taken from the exhibit rather than a later copy of it. Where the verification cannot be reproduced, what remains is the examiner's account of the process, and that is a weaker thing to defend under cross-examination.

What any method achieves on a real handset is governed by the device itself. Chipset, vendor implementation and patch level determine which acquisition types are supported, which makes method selection a decision taken per device rather than set by policy [1]. Fakiha [2] adds the split of user data between the handset and cloud storage, which puts some content beyond device acquisition entirely. Both explain why examiners depend on commercial platforms to establish what is obtainable, and why the NIST Computer Forensics Tool Testing programme matters as the mechanism through which those platforms are checked against something other than their own documentation [5],[6].

2.2 Cellebrite UFED and UFED Physical Analyzer

The Cellebrite platform performs two operations that are often spoken about as one. UFED performs acquisition, negotiating with the handset for a logical, file system or physical image, and what the handset permits is set by its chipset, vendor implementation and patch level rather than by the tool [1]. Physical Analyzer performs decoding, parsing that image and turning application databases into readable conversations, contacts and media. Evaluation has to keep the two apart. An examination can fail because UFED never obtained the data, or because Physical Analyzer obtained it and could not read the format, and those failures need different responses. A report showing no messages for an installed application does not tell the examiner which one happened.

Cellebrite publishes supported-application and version documentation for Physical Analyzer, and examiners use it to judge what an extraction ought to contain. The documentation is vendor-issued and does not come with the test data behind it. Fakiha [2] treats that dependence as a defining constraint on the discipline, because nothing available to the examiner establishes what the tool failed to recover.

Independent testing comes almost entirely from the NIST Computer Forensics Tool Testing programme, which has published results for UFED4PC with Physical Analyzer [5] and for the later Inseyets generation of both components [6]. CFTT works through generic data object categories such as contacts, call logs, SMS and media, recording an anomaly where a category comes back incomplete or wrong. Every cycle records anomalies, and the ones recorded change between cycles, so the platform is not stable across versions even inside the categories that are tested.

Those categories stop where this study starts. CFTT's social media category includes WhatsApp, but it checks account information rather than message content, and it does not cover Signal or Telegram [6]. A clean result for SMS, or for WhatsApp account details, says nothing about whether message content came out. That content sits in application-owned databases in formats the developers do not publish [7], [8].

Comparative studies place UFED beside its competitors without settling the question. Sinaga et al. [11] measure artefact extraction performance and usability across UFED, MSAB XRY and Magnet AXIOM

under logical extraction, and conclude that artefact volume does not by itself reflect evidentiary value, because the differences they record come from artefact classification and provenance rather than from access to core communication data. No application-level result for encrypted messaging appears in the study. Riadi et al. [10] do work at application level, scoring three platforms against Signal, but Cellebrite is not one of the three.

Work using the platform against these applications directly is thin, and narrow in method where it exists. Heath et al. [3] recover WhatsApp and Telegram artefacts with Cellebrite, including content the applications present to the user as deleted, but under logical and advanced logical extraction only. Vasilaras et al. [14] report the sole result located in this review for Cellebrite decoding of Telegram, discussed in Section 2.3, obtained on rooted handsets with full file system acquisition outside their scope. Other studies show the content is reachable without showing that a commercial platform reaches it. Sudiana et al. [13] recover WhatsApp disappearing messages from an unrooted handset, and Son et al. [12] decrypt Signal databases by research means rather than through a forensic tool.

None of this work addresses what the platform does against all three applications at once, under the acquisition method Android now requires.

2.3 Encrypted Messaging Apps

All three applications encrypt content in transit. The differences that matter forensically start after that, in where each one puts its message store, whether that store is protected at rest, and where the key sits. Onik et al. [7] identify local storage design as the variable separating tractable applications from intractable ones.

WhatsApp encrypts in transit under the Signal Protocol [18] and then keeps the decrypted history in a SQLite database inside application-private storage, which a logical request does not reach [1]. The encryption problem is elsewhere, in the backups written to shared storage as .crypt14 archives, which need a key held separately in the application's file directory. Prabowo et al. [8] decrypted those archives on rooted and non-rooted devices alike by extracting the key and matching it to the backup, and found the encryption mechanism consistent across both device states. Possession of the key file is the barrier, not the cipher, and the work was done with WhatsApp-Key-Database Extractor, Binwalk and MOBILedit rather than a forensic platform. The database is also not the only source. Sudiana et al. [13] recovered 83.33 percent of disappearing messages from an unrooted handset under NIST SP 800-101r1 [17], working from backup files and Android notification logs rather than the message store itself. Notification logs sit outside the application altogether, so part of what a WhatsApp examination recovers depends on the operating system rather than on the application or the tool. Heath et al. [3] give the platform figures directly. Cellebrite retrieved 75 percent of the target artefacts before the disappearing timer elapsed and 42 percent after it, against XRY at 50 percent and then zero. Both sets came from logical and advanced logical extraction, so they establish a floor rather than a ceiling for what full file system acquisition might reach.

Signal is harder, because it encrypts the local store as well as the transport. The message database sits under SQLCipher with the passphrase protected by the Android Keystore [12], so an acquisition that copies the database without the Keystore material yields ciphertext. Son et al. [12] decrypted Signal databases by recovering the key through research means rather than through a forensic platform, which separates what is technically possible from what a commercial tool has implemented. Krishnapriya et al. [4] examine what Signal leaves on Android and report a smaller artefact set than an equivalent WhatsApp examination. The clearest measure of tool dependence comes from Riadi et al. [10], who scored three platforms against eleven predetermined parameters and recorded 78.69 percent for Belkasoft Evidence

Center, 26.23 percent for Magnet AXIOM and 9.84 percent for MOBILedit Forensic Express. Cellebrite was not one of the three. The most detailed application-level comparison available for Signal therefore excludes the tool this study evaluates. Signal keeps no server-side message store and no cloud backup by default, so a failed decode leaves nowhere else to look [7].

Telegram makes encryption optional, and one application ends up producing two artefact classes on a single handset. Cloud chats are the default, encrypted between client and server under MTPProto [19] but held on Telegram's own infrastructure in a form the operator can read. Secret Chats are end-to-end encrypted and stay on the originating device. Vasilaras et al. [14] found cache4.db running under WAL journaling with Auto Vacuum and Secure Delete disabled, which makes deleted records recoverable from the write-ahead log but not from the database freelist pages. Recovery from that log is timing dependent. Deleted messages were retrievable immediately after deletion but were erased from the main database once new messages had been committed, which makes the interval between deletion and seizure a variable no examiner controls [14]. Logging out wipes the database completely. Media follows the chat type. Normal chat images could be rebuilt from Android gallery caches and thumbnails outside the application; secret chat attachments could not be recovered once deleted. That study also contains the only Cellebrite result for Telegram decoding located in this review: UFED Physical Analyzer 7.50 decoded normal chat media but not secret, while Oxygen Forensics 14.1 did the reverse, and Magnet AXIOM 5.7 decoded neither where 6.0 decoded normal chat media only. Raza and Hassan [9] map those structures in an Android 8.1 virtual environment, finding text message bodies held unencrypted as binary objects in the messages table of cache4.db while the media content itself stays unreadable. Table names change with the application version, user_contact_v3 becoming user_contact_v7, so a parser written against one release will not necessarily read another.

That Cellebrite result carries two caveats. It came from rooted LG G6 and Samsung A50 handsets running Android 9 and 11, and the study put full file system and physical acquisition outside its scope [14]. So the recovery problem is not one problem repeated three times. WhatsApp turns on acquisition scope and on ancillary operating system sources, Signal on Keystore access, Telegram on chat type and on how recently the write-ahead log was committed. A tool evaluation reporting one completeness figure per application conceals the mechanism that produced it, which is why the encryption architecture analysis in Section 3.2 precedes the results in Chapter 4.

2.4 Prior Tool-Evaluation Studies

Tool evaluations in mobile forensics divide by what the study chooses to count, and the choice determines what the resulting number can be used for. Some count artefacts. Some score recovery against a parameter list the authors define. Some test conformance to published assertions. None of the three produces a figure comparable with the others, which is the first difficulty in reading this literature as a body rather than as a set of separate results.

Conformance testing is the NIST Computer Forensics Tool Testing programme, which runs a populated reference device against documented assertions and records an anomaly wherever a data-object category comes back incomplete or wrong [5],[6]. The method can be repeated and is tied to a named version. What it covers at application level is thin: WhatsApp shows up only as account information, Signal and Telegram not at all [6]. It says nothing about whether message content was decoded.

Counting is what Sinaga et al. [11] do, running logical extraction across two iOS and two Android handsets and recording total artefacts per platform. On the Samsung Galaxy A32 the counts were 28,214 for UFED, 15,007 for XRY and 79,088 for AXIOM, and the authors put the spread down to differences in artefact classification and provenance rather than differences in access to core communication data. Their own conclusion is that artefact volume does not necessarily reflect evidentiary value, which is worth

noting because a reader skimming the counts would conclude the opposite. The same study scored examiner-perceived usability at 71.0 for AXIOM, 69.2 for UFED and 59.7 for XRY on the System Usability Scale, a measure no other study in this review reports.

Parameter scoring is the most common approach and the least portable. Riadi et al. [10] set eleven parameters for Signal and expressed each platform's result as the percentage satisfied. Sudiana et al. [13] worked to NIST SP 800-101r1 [17] and expressed recovery as a percentage of the disappearing messages they had seeded. Heath et al. [3] counted artefacts retrieved before and after a disappearing timer elapsed. Every one of those figures is coherent inside its own study and meaningless outside it, because the author sets the denominator.

Test conditions are the second difficulty. Vasilaras et al. [14] rooted both handsets to reach the Telegram database paths, stated plainly that this is not a forensically sound method for actual evidence, and adopted it because they needed a white-box view. They also put full file system and physical acquisition outside their scope. Heath et al. [3] stayed with logical and advanced logical extraction, and Sinaga et al. [11] with logical extraction alone. The acquisition method Android now requires in practice is missing from the studies that come closest to evaluating it.

Validation is the third. Vasilaras et al. [14] cross-checked their results across more than twenty tools because no single tool's output could be trusted on its own. Yoo et al. [15] show the problem reaching the acquisition itself: a full file system extraction of a live device does not reproduce its hash on repetition, so the conventional integrity check fails at the point where it is most needed.

Read singly these studies report findings. Read together they set conditions for Chapter 3. The parameter set has to be fixed before extraction and held constant across all three applications, or the figures will not compare. Tool and application versions have to be recorded before extraction, because a minor release changes what is decoded [14]. Root status and acquisition method have to be stated, because a rooted device under logical extraction is a different situation from an unrooted device under full file system acquisition, and reporting a percentage without saying which one was used tells the reader almost nothing.

2.5 Legal and Evidentiary Context

An extraction has to satisfy two sets of demands that are often treated as one. Admissibility turns on whether the evidence is what it purports to be and whether the method behind it is reliable. Data protection applies separately, governing law enforcement processing under Directive (EU) 2016/680 whether or not a prosecution follows [16].

Authenticity is conventionally evidenced by hashing the acquired image and reproducing the value on demand, which lets a court be told that the exhibit examined is the exhibit seized. Yoo et al. [15] show the convention does not survive full file system acquisition. A live device keeps writing, repeated extractions return different hashes, and the examiner ends up evidencing integrity file by file and explaining a discrepancy rather than presenting one verified figure.

Reliability of method is a separate question and vendor documentation does not answer it. Fakiha [2] treats dependence on commercial platforms as a structural weakness of the discipline, and the NIST Computer Forensics Tool Testing reports remain the only independent validation evidence available for this platform [5],[6]. For these three applications, they go no further than WhatsApp account details [6].

The Directive adds obligations that operate whether or not anyone reaches a courtroom. Article 4 requires personal data to be accurate and limited to what the purpose needs. Article 6 requires distinction between

categories of data subject, which bites here because a seized handset carries the communications of everyone the user corresponded with, and appearing in an extraction does not make a person a suspect. Article 25 requires logging of collection, alteration, consultation and disclosure in automated systems, which is the statutory form of what chain of custody does in practice.

A tool that returns nothing while indicating nothing threatens both sets of demands at once, since the examiner can neither vouch for completeness nor record what was not obtained.

2.6 Identified Gap

The gap is specific rather than general, and it appears at the point where the studies stop. NIST tests the platform [5], [6], but for these three applications it checks nothing beyond WhatsApp account details [6]. Sinaga et al. [11] set the platform against its rivals and report nothing at application level. Riadi et al. [10] test the applications and leave this platform out. Vasilaras et al. [14] hold the only Cellebrite result for Telegram decoding located in this review, obtained on rooted handsets running Android 9 and 11 with full file system acquisition explicitly outside their scope. Heath et al. [3] use the platform on two of the three applications, under logical and advanced logical extraction only.

What is missing is an evaluation of UFED and Physical Analyzer against all three applications, on a current Android release, under the full file system acquisition that platform encryption now forces, with the parameter set fixed before extraction begins. This dissertation sets out to perform that evaluation, and then examines what silent failure means under Directive (EU) 2016/680 [16].

Chapter 3: Methodology

This chapter explains how the study was built. Section 3.1 sets out the philosophy and the case for a controlled experiment. Section 3.2 works through how each application stores and protects its data, because a low recovery figure means little until the obstacle behind it is known. Section 3.3 describes the device and the corpus seeded onto it. Section 3.4 gives the procedure: which extraction methods were tried, in what order, and what was written down each time. Section 3.5 defines how recovery is scored. Section 3.6 covers ethics and the personal data the extractions captured. Every choice described here was fixed before the first extraction, with one exception: the fifth recovery state in Section 3.5, added once it was clear that no acquisition would reach the application storage. Where the extraction did not follow the plan, this chapter says so and Chapter 4 gives the detail.

3.1 Research Philosophy and Approach

The design is set out here by layer, following Saunders' framework, so that each choice can be judged on its own terms [21].

Philosophy is positivist. A tool either returns a seeded artefact or it does not, and that outcome is observable independently of the examiner. Claims about recovery are therefore tested against a ground truth log rather than interpreted.

Approach is deductive. Vendor documentation, conformance testing and prior application-level research supply expectations about what each application should yield under each acquisition method [5], [6]. The experiment tests those expectations against observed output, and a result below expectation is treated as a finding to be explained through the storage analysis in Section 3.2.

Strategy is Design Science Research, with experiment as the secondary strategy. Design Science Research is the primary strategy because the study does not end in measurement. Its output is an artefact: the practitioner framework in Chapter 6, which states which acquisition methods to attempt against these applications on an unrooted device and what an examiner must record when a method fails or a tool returns nothing [22]. The experiment supplies the evaluation evidence that the artefact rests on [23]. A controlled experiment was preferred to the examination of casework devices, since a casework device carries no record of its own contents and a missing message cannot then be distinguished from a message that never existed. Seeding a device with known content settles that, at some cost to realism. The same reasoning excluded the researcher's existing personal data from the test corpus, since it could not be enumerated completely or examined without exposing third parties.

Methodological choice is quantitative. Recovery is expressed as counts, percentages and hash comparisons against the ground truth log. The legal analysis in Chapter 5 is doctrinal, and it evaluates the artefact rather than generating data, so the data collection itself remains single-method.

Time horizon is cross-sectional. All acquisitions were attempted against one seeded corpus over two sessions, on 16 and 22 September 2026, with application versions recorded before the first attempt [14]. A firmware update installed between the two sessions is reported in Section 4.1. The findings describe those versions at that point and make no claim about behaviour across later releases.

The procedure follows the preservation, acquisition, examination and reporting phases of NIST SP 800-101r1 [17].

3.2 Encryption Architecture Analysis (O1)

Chapter 4 says what came back. This section says what stood in the way. Establishing that first is what allows a low recovery figure to be explained rather than simply stated, and it is the work O1 asks for.

Evidence came from three places. Vendor and platform documentation supplied the protocol and storage claims: the WhatsApp encryption white paper [18], the MTProto specification [19] and Android's file-based encryption documentation [20]. Published forensic analyses supplied what those claims look like on disk, including the WhatsApp crypt14 message store [8], Signal's database and key handling [10], [12] and Telegram's local database [14]. The extraction itself supplied the third, and smallest, part.

Smallest, because the intended source did not materialise. A full file system image would have exposed each application's private data directory for inspection. It was not obtained. Section 3.4 sets out the procedure, and Chapter 4 records the attempts and the errors returned. Inspection was therefore confined to the structure visible in the completed advanced logical extraction. Two alternatives were considered and rejected. Rooting the handset would have granted the access but destroyed the unrooted condition under test. An emulator would have granted neither, since hardware-backed key storage is precisely what an emulator does not reproduce. Encryption claims here are consequently attributed to their sources, and marked unverified where the extraction could not corroborate them.

Four properties were recorded per application: where message and media data is stored, what form it is stored in, what protects it and who holds the key. The last property decides everything else. A database sealed with a Keystore-held key does not open because an examiner has a copy of the file [12], [20].

Storage protection and transport protection were recorded as separate properties. They are frequently treated as one. WhatsApp and Signal encrypt one-to-one messages end to end by default [7], [18]. Telegram does so only in secret chats, and the cloud chats used here are readable by the operator once

they reach its servers [19]. That makes Telegram structurally unlike the other two, and its recovery figures have to be read with that in mind rather than placed alongside theirs as a third comparable case.

Table 3.1 pulls the analysis together. The last two columns state what each method should return if the documented protections work as described. Chapter 4 checks those expectations against what came back.

Table 3.1: Storage and protection properties, with recovery expectations

Application	Message store and form	Protection at rest and key custody	Expected result: Advanced Logical	Expected result: File System, Android Backup (unrooted)
WhatsApp	SQLite database in application-private storage; encrypted .crypt14 backups on shared storage [1], [8]	File-based encryption of private storage [20]; backup key held in the application's file directory [8]	Messages not returned; application data outside logical scope [1]	Message store returned only if the backup route includes application data; backups unreadable without the key [8]
Signal	SQLCipher-encrypted database in application-private storage [12]	Database key protected by the Android Keystore [12]; no cloud backup by default [7]	Messages not returned	A copied database remains ciphertext without Keystore material [12]
Telegram (cloud chats)	cache4.db in application-private storage, WAL journaling; message bodies held unencrypted in the database [9], [14]	File-based encryption of private storage [20]; cloud chats also held on Telegram servers in operator-readable form [19]	Messages not returned	Readable message store if returned; deleted records recoverable only while held in the write-ahead log [14]

No entry in this table could be verified against the extraction, because no completed acquisition reached application-private storage. All entries rest on the sources cited.

Read down the Advanced Logical column and the answer is the same three times. Logical extraction was not expected to reach message content here. Heath et al. did recover WhatsApp and Telegram artefacts under logical and advanced logical extraction [3], but with application versions from 2021, before Android 12 removed the data of applications targeting API level 31 or higher from adb backup [24]. Everything rested on the file system route getting into application-private storage. For Signal, getting in would still not have been enough.

3.3 Experimental Design and Dataset

One handset, one examiner, one corpus. Holding those constant is what allows a difference in recovery to be read as a difference between applications or acquisition methods, rather than a difference between conditions. The subject device was a Samsung Galaxy S25 Ultra (SM-S938B), Android 16 on One UI 8.5, build BP4A.251205.006.S938BXXSBCZG3, unrooted throughout. The corpus was populated on 6 and 7 September 2026, and when the device state was recorded on 11 September, Signal 8.26.4, Telegram 12.10.1 and WhatsApp 2.26.35.75 were installed. Versions were logged because a minor release is enough to change what a tool decodes [14]. WhatsApp and Signal both updated after the corpus was sent, on 8 and 9 September (Figures C.5 and C.3), so the seeded items were written by slightly earlier versions than the ones tested. Versions were not re-recorded at A3 or A4. Rooting was never attempted. It would have delivered access at the price of the condition under test.

Each application carried the same 130 items: 100 text messages and 30 media files (10 photographs, 10 videos and 10 audio clips), 390 items across the study. Message bodies begin with a token of the form APP-DIRECTION-SEQ, so WA-S-001 is the first message sent from the subject device in WhatsApp. This matters more than it may appear. Extraction output frequently arrives stripped of the metadata that would otherwise identify a record, and a token inside the message body survives that.

Deletions came out of the live counts, not in addition to them: ten messages and four media files per application, deleted in one pass once the whole corpus had been sent, each marked a second time as [DEL-WA-S-003]. Deletion was done on 7 September, the day the corpus was completed, so 9 days passed before the first acquisition and 15 before A4. Whether a deleted Telegram record is recoverable turns partly on whether the write-ahead log had been committed by then [14].

Media was generated rather than collected. Photographs ranged from 320x240 to 3024x4032, JPEG and PNG, some with EXIF and some without. Videos ran three to thirty seconds over four resolutions, portrait included. Audio ran three to sixty seconds at 64 to 192 kbps. Every source file was hashed with SHA-256 before it was sent, which turns the integrity question into a comparison rather than a judgement.

Two settings were fixed before anything was sent. Telegram ran in standard cloud chats, not secret chats. Signal ran with normal message types, disappearing messages off. The first of those has a consequence worth stating plainly: only two of the three applications encrypt the messages under test end to end [19]. Chapter 4 reads Telegram's figures against that fact instead of alongside the other two as though the three were equivalent.

Everything sent was entered in a ground truth log holding token, direction, application, item type, send timestamp, deletion timestamp, source hash and a recovery state left blank until analysis. Every claim in Chapter 4 is made against that log. One entry type had to be added during the work. Where no completed acquisition reached the storage holding an item, the item is logged as tested and not acquired. In this study that applied to the whole corpus, as Section 3.5 explains. An item nobody could reach is not the same as an item a tool looked for and missed.

3.4 Data Collection

Acquisition ran on Cellebrite UFED 4PC 7.76.0.24 under the university's Educational licence, and decoding on UFED Physical Analyzer 7.73.0.11, both on a university forensic workstation. Each attempt had its own case identifier and its own folder on a separate local drive. No extraction could overwrite another.

The handset was set up the way UFED's instructions for the SM-S938B require. Developer options, USB debugging and Stay Awake were switched on. USB mode was Transfer files, screen lock was None, and the workstation was authorised for debugging. Samsung Auto Blocker was later switched off as a troubleshooting step (Section 4.4), since it exists to block commands arriving over USB from a connected computer [28]. Build number, patch level and application versions were written down before the first attempt.

The acquisition ladder put the available methods in order of how deep each could reach. Advanced Logical sat at the bottom. File System extraction through Android Backup came next. Physical extraction would have come third, but UFED offered it only as "Physical (Rooted)", and Sections 3.2 and 3.3 have already explained why rooting was ruled out. APK Downgrade was ruled out too. Cellebrite's own description calls it a last resort method during which data alteration will occur [30], and a damaged corpus could not have been rebuilt.

The ladder was worked from the top. File System extraction went first. When it failed, Advanced Logical followed. Each method ran twice. Any troubleshooting change was logged against the attempt that came after it.

Each attempt left the same record: case identifier, start and end time, method, content types selected, and any error message word for word. Every screen was captured, and a selection of those captures makes up Appendix A. UFED wrote a SHA-256 value for each completed extraction at acquisition, and the values are in Appendix B.

Decoding took two passes. Selective application decoding came first, since it lists whichever applications Physical Analyzer has found. A full decode of everything followed. The decoded extraction was then searched for each application name and for the corpus tokens described in Section 3.3, and every corpus item was given a recovery state in the ground truth log.

3.5 Evaluation Criteria

Each corpus item ends up in one of five states.

State	Definition
Fully recovered	Item found with its content, direction and timestamp matching the ground truth log; for media, the SHA-256 hash matches the source file
Partially recovered	Item found but incomplete, such as a message without its timestamp or a media file whose hash does not match
Present but undecoded	Item's storage acquired, but Physical Analyzer did not parse it into a readable record
Absent	Item's storage acquired and examined, and the item not found
Not acquired	Item's storage not reached by any extraction that completed

The fifth state was added during the work. Without it, an item that no extraction could reach would be scored as absent, which implies that the tool examined the storage and found nothing. The two outcomes mean different things in a forensic report and are kept separate here. The addition changes how an unreached item is labelled, not what is measured, and the four measures below were fixed before the first extraction.

Completeness, accuracy and media integrity are calculated per application and per method. Acquisition time is recorded per attempt, since one extraction covers all three applications. Completeness is the percentage of corpus items fully or partially recovered, reported alongside the count of items not acquired so that a low figure is not misread. Accuracy is the proportion of recovered messages whose token, direction and timestamp match the log. Media integrity is the proportion of recovered media files whose SHA-256 hash matches the source. Acquisition time is the interval between the start and end of each attempt. Each attempt is also recorded as completed, partially completed or failed. Each measure is also broken down by artefact type (text message, photograph, video and audio clip), using the item type held in the ground truth log.

Deleted items are scored in the same way and compared with live items from the same application. That comparison depends on live recovery, since a deleted-item rate means little when nothing live was recovered.

3.6 Ethics

The study received university ethics approval on 5 August 2026. All corpus content was synthetic and was exchanged only between test accounts controlled by the researcher.

The handset was the researcher's own device, so the extractions also captured the researcher's personal SMS, contacts and account data, which include information about third parties. That data is processed under the research provisions of the General Data Protection Regulation [33]. Examination of it was limited to keyword searches for application names and corpus tokens. No personal content was read beyond those searches or reported. Extraction files are held on an encrypted drive and will be securely deleted on completion of the dissertation, once final marks have been confirmed. Device identifiers and account details are redacted from every screenshot reproduced in Appendix A.

Chapter 4: Results and Troubleshooting

Section 3.4 set out an acquisition ladder on the assumption that at least one rung would reach the private storage of WhatsApp, Signal and Telegram. Recovered items could then be scored against the ground truth log. On the test device, no rung got that far. Four acquisition attempts were made over two sessions. Both File System attempts through Android Backup failed with the same error. The two Advanced Logical attempts finished, but neither returned anything from the three applications. This chapter sets out what happened in each attempt, what changed between attempts, and what the decoded extractions contained. The final section explains why the ladder stopped at the logical rung.

4.1 Acquisition Log

Tools, workstation and recording followed Section 3.4, and every captured screen is in Appendix A with device identifiers redacted. UFED identified the handset as a Samsung SM-S938B/DS Galaxy S25 Ultra and reported it as not rooted. At the start of A3 it counted 168 installed applications and 238 GB of 512 GB in use. All 390 corpus items had been on the handset since 7 September 2026, nine days before the first acquisition.

Table 4.1 lists the four attempts. A3 and A4 times come from their extraction summaries. A1 and A2 never produced an extraction file, so their times come from the screenshots taken when the method was selected and when the error appeared. They are accurate to the minute. Outcomes use the categories in Section 3.5.

Table 4.1: Acquisition log

ID	Date	Extraction type and method	Case opened	Extraction	Duration	Outcome
A1	16/09/2026	File System, Android Backup	11:11	11:26 to 11:36	c. 10 min	Failed: "ADB backup failed. Shared memory was partially extracted or failed"
A2	16/09/2026	File System, Android Backup	11:42	12:01 to 12:40	c. 39 min	Failed: same error as A1
A3	16/09/2026	Advanced Logical, all content types	12:45	13:31:06 to 14:00:07	29 min	Partially completed: same ADB error raised during Files
A4	22/09/2026	Advanced Logical, all content types	Not recorded	11:38:42 to 12:25:24	47 min	Partially completed: same ADB error raised during Files; 8,543 files returned

The software state did not stay fixed. On 16 September UFED read the security patch level as 2026-07-05. On 22 September Physical Analyzer read it as 2026-08-05, with build S938BXXSCCZH1. The handset had updated its firmware between A3 and A4. That made no difference to the result, because A3 and A4 failed at the same stage with the same message. It still breaks the fixed software state the design assumed. The handset should have been kept isolated from networks between sessions as well as during them. Section 5.2 and Chapter 7 return to this.

UFED wrote a SHA-256 value for each completed extraction as it saved it. Table 4.2 gives the values for the two extraction archives. Appendix B holds those and the values for the summary reports. The A3 and A4 hashes do not match, and they should not. These are two acquisitions of a live handset that kept writing in between, and a second extraction of the same device does not reproduce the first one's hash [15]. Each value vouches for its own extraction, not for the device. A1 and A2 left no File System image, so there was nothing to hash.

Table 4.2: SHA-256 values recorded by UFED at acquisition

ID	Extraction file	SHA-256
A3	Samsung GSM_SM-S938B_DS Galaxy S25 Ultra.zip	A853A2D9D3E9BC4E83E721E16E6EF53B012BE6F0087DFB31B3651B5AFB1B1D70
A4	Samsung GSM_SM-S938B_DS Galaxy S25 Ultra.zip	EF0083D0CA866DE3D6D00C7CDAC51094A0F212DA41836B091E5B3E98435FBF3E

4.2 File System (Android Backup) Attempts

File System extraction through Android Backup was the second rung, and the one expected to reach the application sandboxes. Full File System extraction was not offered for this handset on this licence (Figure A.2). Once it had detected the handset, UFED offered six extraction types: Advanced Logical, File System, Physical (Rooted), Camera, Screenshot and Chat Capture. Under File System it listed two modes, Android Backup and Android Backup APK Downgrade. It listed no agent-based or exploit-based method for the SM-S938B on this licence. Camera, Screenshot and Chat Capture record what the handset displays rather than extracting stored data. They sit at the level of manual examination, outside the extraction and decoding under evaluation, and were not used. Physical needed root, and Section 3.4 has already explained why that was ruled out.

A1 used Android Backup. UFED started reading the backup stream and showed a file transfer counter. About ten minutes in, it stopped with "ADB backup failed. Shared memory was partially extracted or failed". Straight after the error it asked for the screen timeout to be set back to its original value. A tool that has lost its connection does not do that. UFED had ended the session with the handset in the normal way.

A2 used the same method under a new case identifier, once the measures in Section 4.4 were in place. It ran for about 39 minutes, roughly four times as long as A1, and ended with the same message. The longer run probably means more of the shared storage was read before it stopped. The end state was no different. The error message points to the two halves of the Android Backup method. One half asks Android for an application backup through adb backup. The other copies shared storage, the user-visible area with folders such as DCIM and Download. The message says shared memory was only partly extracted. Neither attempt left a usable File System image.

APK Downgrade was not tried, for the reason Section 3.4 gives. The method swaps an installed application for an older version that still allows backup, takes the backup, then puts the current version back, and Cellebrite states that data alteration will occur [30]. The handset held a corpus that could not be rebuilt in the time left. Risking the data under test for a chance of recovering it was not justified.

4.3 Advanced Logical Attempts

Advanced Logical ran twice, with every available content type selected. Cellebrite describes it as one workflow that combines logical extraction with the ADB backup form of file system extraction [29]. Physical Analyzer bears this out, recording the A4 extraction type as "Advanced Logical [Android Backup]". So Advanced Logical leans on the same backup mechanism that failed in A1 and A2. A completed Advanced Logical run says nothing about whether that mechanism worked.

The A3 case was opened at 12:45 on 16 September, on the same firmware as A1 and A2. Extraction ran from 13:31 to 14:00. On the content selection screen, Email, IM, Browsing Data and User Dictionary were already greyed out as unavailable. Selecting Files brought a warning that it "may require a device restart during the extraction process", and the extraction was allowed to continue. Selecting Contacts brought a list of the accounts UFED could read: the phone contacts, several Google accounts and one WhatsApp account.

In both A3 and A4 the Files stage threw the same "ADB backup failed" message as the File System attempts. Both times UFED carried on, marked the other content types complete and reported the extraction as successful. A4, run six days later on the updated firmware, reproduced A3. Table 4.3 gives the A4 summary.

Table 4.3: Advanced Logical (A4) extraction summary

Content type	Result	Content type	Result
Contacts	911	SMS	8,116
MMS	180	Calendar	289
Pictures	237	Videos	8
Audio/Music	0	Ringtones	0
Documents	1,818	Archives	5
Call logs	1,998	Files	8,543 files
Locations	Completed	Advertising ID	Completed
Email	Not supported	IM	Not supported
Browsing data	Not supported	User dictionary	Not supported

As a system-level extraction, A4 was productive. It holds thousands of SMS, contacts and call records and several thousand files from shared storage. IM, the content type that would carry messaging data, was "Not supported" before the run began. The WhatsApp entry among the contact accounts is worth a second look. Android tells an outside tool that a WhatsApp account exists, through its contacts provider. It does not hand over the messages. The logical rung reached the edge of each application and no further.

4.4 Troubleshooting Applied

Every setting that might plausibly cause an ADB failure was checked or changed. Table 4.4 lists the measures in place for the later attempts, including those set during preparation (Section 3.4).

Table 4.4: Troubleshooting measures

Measure	Purpose
Auto Blocker turned off	Samsung Auto Blocker blocks commands passed through USB cables when a PC is connected [28]

USB mode set to File Transfer	UFED's device instructions require the "Transfer files" mode for multimedia transfer
Screen timeout set to 10 minutes and Stay Awake enabled	Keeps the screen active for the full extraction, as UFED's device instructions require
Different USB port and cable	Rules out a faulty connection as the cause of the interrupted transfer
Antivirus paused on the workstation	Prevents real-time scanning from locking files as they are written
Phone and PC restarted	Clears stale ADB sessions and background processes that compete for the device

Auto Blocker was the prime suspect. Samsung built it to stop commands and software passing over USB [28], which is exactly what an ADB-based extraction does. Turning it off changed nothing.

Nothing else did either. A2, A3 and A4 all ran with the measures in place and failed at the same point as A1. By then the cable, the port, the workstation state and every relevant device setting had been changed, so the connection was not the cause. UFED also kept talking to the handset after each error. It read device properties and finished the other content types, which a broken connection would have prevented. That leaves one explanation. The handset answered the backup request with less data than UFED expected.

The supervisor reviewed the error messages and screenshots. In a verbal discussion he advised that the failure comes from the security design of the Galaxy S25 Ultra, not from how the extraction was run, and that it should be treated as a limitation of the study rather than pursued with riskier methods. The documentation reviewed in Section 4.6 supports that view.

4.5 Decoding and Recovery Scoring

A4 was loaded into Physical Analyzer with selective application decoding. The selection screen offered three packages, all Android system components, and reported "0 of 3 apps decoded by Cellebrite". WhatsApp, Signal and Telegram were not on the list. This was not a decoding failure. The extraction held nothing from the three messengers for the decoder to work on.

A full decode followed. The Analyzed Data view showed 8,294 instant messages (8,115 phone and 179 native), 8,018 media items, 1,998 calls and 897 contacts. The instant message count comes from Android's own SMS and MMS store, not from any third-party application. Physical Analyzer's totals differ slightly from UFED's in Table 4.3, by one SMS, one MMS and 14 contacts. The two tools count at different stages, acquisition and decoding, and the difference has no bearing on the findings. Under Data Files it listed 12 application files, 2,930 documents and 653 uncategorised files. A keyword search of the whole extraction for "WhatsApp", "Signal" and "Telegram" returned nothing. Nor did a second search for the corpus message tokens and media labels. Not one of the 390 seeded items turned up in any decoded category. The UFED summary report for A4 records the device's live encryption state as file-based and the number of installed applications captured as zero, although UFED had counted 168 applications on the handset when it was detected.

Of the five states in Section 3.5, only one applies. All 390 items are not acquired, because no completed extraction reached the storage that held them. Calling them absent would say the tool looked and found nothing. It never looked. Table 4.5 gives the scores.

Table 4.5: Recovery scoring against the ground truth log

Application	Items in corpus	of which deleted	Fully recovered	Partially recovered	Present, undecoded	Absent	Not acquired
WhatsApp	130	14	0	0	0	0	130
Signal	130	14	0	0	0	0	130
Telegram	130	14	0	0	0	0	130
Total	390	42	0	0	0	0	390

The other measures in Section 3.5 follow from this. Completeness is 0 percent for every application, artefact type and method, with all 390 items not acquired. Accuracy and media integrity cannot be calculated, since nothing was recovered to check against the log or the source hashes. Acquisition time is in Table 4.1. The deletion variable could not be tested either. The 42 deleted items sit inside the 390, and a deleted-versus-live comparison needs at least one live item recovered first.

4.6 Interpretation

For all three applications the ladder stopped at the logical rung. The published documentation explains why, and it points to several separate barriers. Any one of them would have been enough.

The backup mechanism is the first. Since Android 12, adb backup leaves out the data of any application that targets API level 31 or higher, unless the application is marked debuggable [24]. Google Play has required new applications and updates to target API level 35 since 31 August 2025 [26]. All three messengers came from Google Play and had been updated in the month before testing, so all three fall under that rule. Every extraction method UFED offered for this handset goes through adb backup. Even a clean transfer could not have carried their data. Cellebrite says as much in its own guidance: on newer devices with newer operating systems, an examiner will not get access to third-party applications such as WhatsApp [29].

The applications can also opt out for themselves. Android lets an application exclude itself from backup by setting `android:allowBackup` to false [25], and WhatsApp is reported to do this [31]. That rule is older than the Android 12 change, and it would have kept WhatsApp out regardless.

Storage encryption sits underneath both. Any device launched with Android 10 or later must use file-based encryption, which keeps application data in credential-encrypted storage until the user unlocks the device [20]. Samsung derives the keys inside Arm TrustZone and Knox Vault and binds them to the user's credential and a device secret [27]. Reading application storage directly needs privileged access, and an unrooted handset does not give it. WhatsApp keeps its decrypted message database in application-private storage, out of reach of a logical request [1], and holds the key to its encrypted backups in its own file directory [8]. Signal goes a step further and encrypts its database with SQLCipher under a key held by the Android Keystore [12]. For Signal, a successful file copy would still have produced ciphertext. Each of these outcomes was predicted in Table 3.1.

This is why the troubleshooting in Section 4.4 could not work. Every measure targeted the link between workstation and handset, and the link was never the problem. The limit is in the operating system and in the applications, which matches the supervisor's reading.

The finding is narrow, but it fills a real gap. A 2026 systematic review found that 58 percent of Android anti-forensics studies did not state the Android version tested, and that none of the experimental studies it covered used Android 12, 15 or 16 [32]. Survey work notes that logical extraction can be limited by device safety features and encryption [2]. It does not say where a current commercial tool stops on a current flagship. This chapter records that point for one device, one build and one tool version. NIST SP 800-101r1 ranks logical and file system extraction below physical methods in both invasiveness and yield [17]. On this handset the two lower rungs returned system data only. Reaching application data would have needed root access, APK Downgrade or, possibly, a higher licence tier. None of these was available within the study's ethical and practical limits.

Telegram is a partial exception. Its standard cloud chats are not end-to-end encrypted, and Telegram holds them on its servers [19]. On the handset Telegram behaved like the other two applications, but the evidential consequence is different. A Telegram cloud chat that cannot be taken from the device may still be obtainable from Telegram through a lawful request. WhatsApp and Signal messages cannot be obtained from their providers in readable form. Chapter 5 picks up this difference.

Chapter 5: Legal and Evidential Implications

Chapter 4 ended with 390 items not acquired. An examiner still has to put that result in front of a court, and this chapter asks what it can bear. Section 5.1 looks at admissibility: whether an acquisition that failed, or only partly succeeded, can support a finding at all, and what the examiner has to disclose. Section 5.2 tests the chain of custody against a log that holds two failed attempts, a firmware change mid-study and two extractions whose hashes differ. Section 5.3 sets the results against Directive (EU) 2016/680, which governs how law enforcement processes personal data [16]. Section 5.4 covers the ethical questions, among them the third-party data an extraction picks up whether the examiner wants it or not.

5.1 Admissibility

There is no EU law on the admissibility of evidence in criminal trials. Collection, use and admission are left to each Member State's criminal procedure [34]. Article 82(2) TFEU would allow a directive on mutual admissibility, but none has been adopted [35]. The European Court of Human Rights does not exclude irregularly obtained evidence as a rule. It asks whether the proceedings, taken as a whole, were fair [34]. Admissibility of the Chapter 4 results is therefore a national question. The criteria that keep recurring across these frameworks can still be applied to them.

The European Law Institute's model directive, written as a blueprint for future EU legislation, requires safeguards for the integrity, authenticity and completeness of electronic evidence. It also gives the defence a right to challenge it [35]. For what they contain, A3 and A4 meet the first two. Each carries a SHA-256 value that UFED recorded at acquisition (Table 4.2), and every step is on screen in the captured images. Completeness is the problem. UFED reported both Advanced Logical runs as successful, and neither held a single item from the three messaging applications. An examiner reading only the extraction summary would have had no sign that the application data was never reached.

That turns the distinction in Section 3.5 between absent and not acquired into an evidential question, not only a methodological one. "No WhatsApp, Signal or Telegram messages were found" is true of the extraction. Said of the handset, it misleads. ENFSI guidance requires the use of unvalidated methods, every known error and the uncertainty of the methods to be declared [36]. On this device the uncertainty is complete. The method returned nothing, and could not have returned anything.

The same guidance says tool functions cannot be validated on their own, because the setting a tool is used in is part of the process [36]. NIST has tested the Cellebrite platform against generic data categories [5], [6]. It has not tested it against these applications on this handset. Chapter 4 is exactly that kind of in-situ check, and it shows the platform falling short. A court weighing overall fairness would expect that limitation to be stated, not left for someone else to work out.

5.2 Chain of Custody

Section 2.1 defined chain of custody as the continuous record of who held the device, when, and what was done to it. The handset never left the researcher's possession, and each acquisition attempt has its own case identifier and a screenshot of every step. The record is continuous. What it records raises three problems an examiner would have to answer.

Attempts that produced nothing are the first. A1 and A2 left no image and no hash, but both were still actions performed on the exhibit. A custody record listing only the completed extractions would misstate how the device was handled. Every extraction also changed the handset. UFED's instructions had the screen lock removed and the screen timeout extended, and Physical Analyzer records that a client application was used for extraction. None of this is unusual for a logical acquisition. It still alters the device, and it has to be declared.

The firmware update between A3 and A4 is the second. The handset was put in airplane mode for each extraction session and taken out of it when the day's work was done. It picked up the update in between. NIST guidance calls for a device under examination to be kept isolated from networks for exactly this reason [17]. Isolating it only while an extraction runs does not stop it changing between extractions. The update made no difference to the result, since A3 and A4 failed the same way. In casework it would mean the two extractions describe two different versions of the device's software.

Then there are the hashes. A3 and A4 have different SHA-256 values. Yoo et al. show that repeat extractions of a live device do not reproduce the same hash, because the device keeps writing in between, and that integrity then has to be shown one extraction at a time [15]. That is what Table 4.2 does. Each value vouches for its own extraction. ENFSI guidance requires the examiner to show that the data analysed is a valid copy of the original [36]. A recorded hash meets that for each extraction. It does not meet it for the device as a whole.

The custody record, then, is complete for what was obtained. Its gaps lie in what the device did during and between acquisitions, and those gaps are stated rather than left out.

5.3 Jurisdictional Considerations

Extracting data from a seized handset is processing of personal data by a competent authority. Inside the EU, that falls under Directive (EU) 2016/680 [16], and three of its requirements bear directly on Chapter 4.

Article 4 requires personal data to be accurate and limited to what the purpose needs. A4 returned 911 contacts, 8,116 SMS and several thousand files, almost none of them to do with the applications under examination. The data that did relate to them was never obtained. Accuracy reaches the report too. Logging the messaging applications as empty, rather than as not acquired, would put inaccurate information on the case record. Article 6 requires different categories of data subject to be kept apart, and most of the contacts and SMS captured here belong to people with no link to any investigation. Article 25

requires processing to be logged. Table 4.1, failed attempts included, is the kind of record that article has in mind.

The results also point past the device. Regulation (EU) 2023/1543, applicable from 18 August 2026, lets competent authorities send European Production Orders to providers of interpersonal communication services offered in the Union, covering subscriber, traffic and content data [37]. Telegram holds standard cloud chats in a form it can read [19], so an order could in principle return what the handset would not. WhatsApp and Signal encrypt end to end and hold no readable copy [7], [18]. An order to either could not produce message content.

5.4 Ethical Issues

Section 3.6 covers the ethics of the study itself. This section deals with the questions an examiner in the same position would face in casework.

Pressure to escalate comes first. Once the available methods fail, what remains are the methods that change the evidence. Cellebrite calls APK Downgrade a last resort during which data alteration will occur [30]. Vasilaras et al. state that the rooting they used to reach Telegram's database is not a forensically sound method for actual evidence [14]. Either route trades the integrity of the exhibit for a chance of recovery. That is not the examiner's call to make alone. It belongs with the investigating and prosecuting authorities, made on a clear statement of what could be lost.

Then there is the null result. UFED called both Advanced Logical extractions successful. The examiner's duty runs to the court, not to the tool's summary screen, and ENFSI guidance requires known errors and the uncertainty of a method to be declared [36]. "No messages found", with no word that the messages were never within reach, would be accurate in form and misleading in substance.

Collateral intrusion is the last. The extraction swept up the contacts and SMS of hundreds of people with no connection to any investigation, and recovered nothing from the applications it was aimed at. Directive (EU) 2016/680 limits data to what the purpose needs [16]. UFED lets content types be chosen before extraction starts. An examiner who expects messaging data to be out of reach should narrow the extraction to match, not collect everything by default.

Chapter 6: Discussion and Practitioner Framework

Chapter 4 said what happened on the handset. Chapter 5 said what that means in front of a court. This chapter turns both into the artefact that Section 3.1 promised. Section 6.1 draws the results together across the three applications. Section 6.2 sets out the practitioner framework: which acquisition methods to try on an unrooted Android handset, when to stop, and what to record when a method fails or returns nothing. Section 6.3 sets the findings against the literature in Chapter 2.

6.1 Synthesis

The three applications ended in the same place. All 390 corpus items were not acquired, and the reason was the same each time. Every extraction method UFED offered for this handset ran through the Android backup route, and that route does not carry the data of current messaging applications [24], [25]. The differences set out in Table 3.1 never came into play. Signal's SQLCipher database, WhatsApp's backup key and Telegram's write-ahead log would each have shaped recovery once the file system was reached. It never was. RQ2 is answered, then, though not in the form it expected. On an unrooted Android 16

handset, with the licence available to this study, UFED and Physical Analyzer recovered nothing from any of the three applications, and the comparison between them could not be made.

Three findings come out of that. The ladder stops at the logical rung, and every rung above it needs root, a destructive method or, possibly, a higher licence tier. The tool's own status cannot be trusted on its own. UFED marked both Advanced Logical runs successful despite the backup error, and only the application list in Physical Analyzer showed that nothing had come from the three messengers. An examiner reading the status alone would have reached the wrong conclusion. Telegram, finally, is the same on the device and different off it. Its standard cloud chats stay on the provider's servers in readable form [19], which leaves a route open that is closed for WhatsApp and Signal.

Those findings set the brief for Section 6.2. The framework has to tell an examiner when to stop, how to confirm a failure rather than take a status at face value, and where to go once the device has nothing more to give.

6.2 Proposed Framework

This is the artefact Section 3.1 committed the study to. Its reach is deliberately narrow: end-to-end encrypted messaging applications on unrooted Android handsets, examined with a commercial platform whose available extraction methods all run through the Android backup route. That is what was tested, and the framework claims no more. It comes in two parts. Table 6.1 gives the five stages and the condition for moving on from each. Table 6.2 lists what to record at every attempt, whether it succeeds or not.

Table 6.1: Acquisition decision framework for end-to-end encrypted messaging applications on unrooted Android handsets

Stage	Action	Proceed or stop	Basis
1. Isolate and record	Keep the handset isolated from all networks from seizure until the final extraction, not only during each session. Record model, build number, security patch level, root status and the version of each target application.	Proceed once the device state is recorded	[17]; Section 5.2
2. Scope and acquire	Select only the content types the purpose requires. Attempt the deepest non-destructive method the tool offers for the model, then step down on failure.	Proceed to Stage 3 whatever the tool reports	[16]; Sections 4.2–4.3
3. Verify independently	Confirm that each target application appears in the decoding list. Search the extraction for the application names and any known identifiers. Classify each attempt as completed, partially completed or failed, and each item as absent or not acquired.	If application data is present, examine it. If not, go to Stage 4	[36]; Sections 3.5, 4.5
4. Stop or escalate	Do not use destructive methods (APK Downgrade, rooting) on the examiner's own judgement. Escalate only on a documented decision by the investigating or prosecuting authority, stating the risk of alteration.	Default: stop device acquisition	[14], [30]; Section 5.4

5. Look beyond the device	Consider a request to the service provider. Content can be obtained only where the provider holds it in readable form, as with Telegram cloud chats. For end-to-end encrypted services, expect subscriber and traffic data at most.	End	[7], [18], [19], [37]; Section 5.3
----------------------------------	---	-----	------------------------------------

Stage 3 is the core of the framework. Chapter 4 showed that UFED reported both Advanced Logical extractions as successful even though neither contained any data from the target applications. The framework therefore treats the tool's status as one input rather than a conclusion, and requires the examiner to confirm the outcome independently before recording it. Stage 4 sets the default at stopping. Destructive methods remain available, but the decision to use them is moved from the examiner to the authority responsible for the case, where the loss of evidential integrity can be weighed against the value of what might be recovered.

Table 6.2: Fields to record at each acquisition attempt

Field	Why it is recorded
Device state (model, build, patch level, root status, network isolation)	Establishes which software state the extraction describes (Section 5.2)
Application name and version for each target	A minor release can change what a tool decodes [14]
Attempt identifier, case identifier, tool and version	Separates attempts and ties each to a reproducible toolset
Method, mode and content types selected	Shows what was attempted and what was deliberately excluded [16]
Start and end time	Acquisition time; places the attempt in sequence
Exact error text, including errors the tool continued past	A completed status can hide a failed stage (Section 4.3)
Changes made to the device (settings altered, client installed)	Declares actions performed on the exhibit (Section 5.2)
SHA-256 of each output file	Verifies each extraction on its own terms [15]
Tool-reported status and independently verified status	Records any disagreement between the two (Section 4.5)
Applications present in the decoding list	Distinguishes not acquired from absent (Section 3.5)
Decision not to escalate, or the authority's decision to escalate, with reasons	Makes the Stage 4 choice reviewable [36]

Each field in Table 6.2 answers something that happened in Chapter 4 and would otherwise have gone unrecorded or been misread. Taken together they make up a processing log of the kind Article 25 of Directive (EU) 2016/680 requires [16], and they support the declaration of known errors and uncertainty that ENFSI guidance expects [36].

The traceability matrix in Section 1.8 set two criteria for judging the framework. The first is coverage of the acquisition ladder. Every rung offered for the test device is addressed, the two the framework advises against included. The second is coverage of what to log when a method fails or returns nothing. Every field traces back to a failure point observed in Chapter 4. The framework's limits are the study's. It rests on one device, one build and one tool version, and no practitioner has yet tested it.

6.3 Discussion against the Literature

The nearest prior studies recovered data. This one recovered none. Heath et al. retrieved 75 percent of the target artefacts with Cellebrite before the disappearing timer elapsed, under logical and advanced logical extraction [3]. Sudiana et al. recovered 83.33 percent of WhatsApp disappearing messages from an unrooted handset [13]. The gap fits the Android 12 change to adb backup [24], which took application data out of the route both methods rely on. Proving it would mean comparing the device and application versions each study used. Sudiana et al. also worked from backup files and Android notification logs rather than the message store [13]. Notification logs were not examined here, and they sit outside the application entirely.

Sinaga et al. concluded that artefact volume does not by itself reflect evidentiary value [11]. A4 bears that out in its strongest form. Thousands of SMS, contacts, call records and files came back, and nothing from the three applications under examination. Judged by volume, the extraction looked productive.

NIST validates the platform against data categories, and for these applications it goes no further than WhatsApp account details [5], [6]. Section 2.2 warned that a clean category result proves nothing about message content, and A4 shows why. It returned the WhatsApp account through the contacts provider, which is about as far as CFTT looks, and not one WhatsApp message. By category the extraction looked largely complete. By application it was empty.

Vasilaras et al. hold the only Cellebrite result for Telegram decoding found in this review. It came from rooted handsets on Android 9 and 11, and the authors state that rooting is not a forensically sound method for actual evidence [14]. On an unrooted Android 16 handset the Telegram database was never reached. Their result rests on an access condition casework cannot normally meet. Son et al., who decrypted Signal databases by research means [12], and Riadi et al., who scored three other platforms against Signal [10], share that dependence. Both start from a database already acquired. On this device, it could not be.

Two findings in the literature hold up. Yoo et al. showed that repeat extractions of a live device do not reproduce the same hash [15], and A3 and A4 do exactly that. Dowrick et al. found no experimental study on Android 12, 15 or 16 in the anti-forensics literature they reviewed [32]. This study adds one, though what it records is a limit, not a recovery rate.

Section 2.6 defined the gap as an evaluation of UFED and Physical Analyzer against all three applications, under full file system acquisition, on a current Android release. That evaluation could not be run as designed. What the study shows instead is that, on this device and this licence, full file system acquisition was not available. For a practitioner, that is the more pressing finding.

Chapter 7: Conclusion

7.1 Summary per Objective

The study asked how much of WhatsApp, Signal and Telegram Cellebrite UFED and Physical Analyzer recover from an Android 16 handset, and what follows when a tool falls short of its documented capability. Each objective is taken in turn below.

O1 asked what had been tested, what had not, and how each application stores and protects its data. Chapter 2 located the gap. Section 3.2 set out each application's storage and protection in Table 3.1, with an expectation for each acquisition method. Chapter 4 bore out the Advanced Logical expectations. The File System expectations could not be tested, because that route never completed.

O2 asked for a controlled extraction across all three applications on identical seeded data. A 390-item corpus was populated and logged, and four acquisition attempts were made under the procedure in Section 3.4. The design held. The device did not allow it to be completed. Both File System attempts through Android Backup failed. Both Advanced Logical attempts completed without reaching application data.

O3 asked for tool performance to be measured and compared. Completeness was 0 percent for all three applications under every method, with every item not acquired. Accuracy and media integrity could not be calculated. Acquisition times were recorded. The comparison between applications could not be made, because acquisition stopped short of the point where the applications differ.

O4 asked for the admissibility and custody consequences under Directive (EU) 2016/680, and for a practitioner framework. Chapter 5 made the assessment. Section 6.2 presents the framework in two tables: a five-stage decision process, and a checklist of what to record when a method fails or returns nothing.

On the second half of the aim, the tool did not fall short of its documentation. Cellebrite's own guidance says third-party application data is out of reach on newer Android devices [29], and that is what happened. The shortfall was in what the tool reported. UFED marked both Advanced Logical runs as successful, and nothing in its summary showed that the messaging data had never been reached. Chapters 5 and 6 deal with what follows from that.

7.2 Contribution

The main contribution is a documented limit. On an unrooted Samsung Galaxy S25 Ultra running Android 16, Cellebrite UFED 7.76 recovered nothing from WhatsApp, Signal or Telegram under any extraction method the study had access to. A 2026 systematic review of Android anti-forensics research found no experimental study on Android 12, 15 or 16 among the 78 it covered [32]. This study supplies one data point, with the build, application versions, error text and hash values anyone would need to repeat it or dispute it.

Method gains a new recovery state. "Not acquired" separates what a tool could not reach from what it looked for and missed, and Chapter 5 showed that the difference carries evidential weight. The study also shows that a tool's completion status cannot be taken at its word. UFED reported success on two runs that held nothing from the target applications.

Practice gains the framework in Section 6.2. It tells an examiner when to stop, how to check a result without relying on the tool's own report, and what to write down when a method fails or comes back empty.

7.3 Limitations

Everything here rests on one handset, one tool version and one licence. The UFED installation available to the study, licensed at Educational level, offered no agent-based or exploit-based method for this model. Whether a higher tier would have done so was not established, so the findings describe this installation, not the Cellebrite platform as a whole.

The software state did not remain fixed. The handset was isolated during each extraction session but not between sessions, and a firmware update arrived between A3 and A4. It did not change the result. It did

break the single software state the design assumed. Application versions were recorded once, on 11 September, and were not checked again after the handset went back online between sessions.

Some sources went unexamined. Android notification logs, which Sudiana et al. used to recover WhatsApp messages from an unrooted handset [13], sit outside the application and were not looked at here. Cloud and provider-held data were out of scope by design.

With nothing acquired, three planned comparisons could not be run: deleted against live items, media against messages, and one application against another. RQ2 is answered only in part.

A1 and A2 timings come from screenshot timestamps and are good only to the minute. The corpus was seeded, not real, and seeded conversations are tidier than real ones, even though the handset itself was in daily use. The framework in Section 6.2 has been judged against the criteria in Section 1.8, but no practitioner has yet used it in casework.

7.4 Future Work

The obvious next step is to run the same experiment on the same model with a higher Cellebrite licence tier, and with other platforms such as XRY and AXIOM. That would show whether any commercial method reaches application data on an unrooted Android 16 handset. A rooted handset of the same model, seeded with the same corpus, could then measure the application-level differences in Table 3.1 that this study never reached. Notification logs and other operating system sources should be examined as another route to message content. The framework in Section 6.2 needs testing by examiners on real cases, and revising in the light of that use. And since every Android release and application update can shift what a tool reaches, this kind of study is worth repeating at intervals rather than treating as settled.

References

- [1] A. Almuqren, H. Alsuwaelim, M. M. H. Rahman, and A. A. Ibrahim, "A systematic literature review on digital forensic investigation on Android devices," *Procedia Computer Science*, vol. 235, pp. 1332–1352, 2024.
- [2] B. Fakiha, "Unlocking digital evidence: Recent challenges and strategies in mobile device forensic analysis," *Journal of Internet Services and Information Security*, vol. 14, no. 2, pp. 68–84, 2024.
- [3] H. Heath, A. MacDermott, and A. Akinbi, "Forensic analysis of ephemeral messaging applications: Disappearing messages or evidential data?," *Forensic Science International: Digital Investigation*, vol. 46, art. 301585, 2023.
- [4] S. Krishnapriya, V. S. Priyanka, and S. Satheesh Kumar, "Forensic extraction and analysis of Signal application in Android phones," in *Proc. 2021 Int. Conf. Forensics, Analytics, Big Data, Security (FABS)*, 2021, doi: 10.1109/FABS52071.2021.9702702.
- [5] National Institute of Standards and Technology, "Test results for mobile device acquisition tool: UFED4PC v7.60.0.222 – Physical Analyzer v7.58.0.66," U.S. Department of Homeland Security Science and Technology Directorate, Feb. 2023.
- [6] National Institute of Standards and Technology, "Test results for mobile device acquisition tool: Cellebrite Inseyets.UFED v10.3.0.260 – Inseyets.PA v10.3.0.3169," U.S. Department of Homeland Security Science and Technology Directorate, Sep. 2024.
- [7] A. R. Onik, J. Brown, C. Walker, and I. Baggili, "A systematic literature review of secure instant messaging applications from a digital forensics perspective," *ACM Computing Surveys*, vol. 57, no. 9, art. 239, May 2025, doi: 10.1145/3727641.
- [8] W. A. Prabowo, F. Mohsen, and S. R. Selamat, "WhatsApp mobile applications in the lens of digital forensics: Deciphering the Msgstore.db.crypt14 file," *Journal of Cyber Security and Mobility*, vol. 14, no. 4, pp. 823–848, 2025, doi: 10.13052/jcsm2245-1439.1443.
- [9] A. Raza and M. B. Hassan, "Digital forensic analysis of Telegram Messenger app in Android virtual environment," *Mobile and Forensics*, vol. 4, no. 1, pp. 31–43, 2022, doi: 10.12928/mf.v4i1.5537.
- [10] I. Riadi, Herman, and N. H. Siregar, "Mobile forensic analysis of Signal messenger application on Android using Digital Forensic Research Workshop (DFRWS) framework," *Ingénierie des Systèmes d'Information*, vol. 27, no. 6, pp. 903–913, 2022, doi: 10.18280/isi.270606.
- [11] S. J. Sinaga, M. A. Asykar, H. Manurung, W. C. Wibowo, S. Yazid, and T. O. Edwardo, "Comparative evaluation of artifact extraction performance and usability in digital forensic tools: A study

of Cellebrite UFED, MSAB XRY, and Magnet AXIOM," *Journal of Forensic Sciences*, early view, 2026, doi: 10.1111/1556-4029.70320.

[12] J. Son, Y. W. Kim, D. B. Oh, and K. Kim, "Forensic analysis of instant messengers: Decrypt Signal, Wickr, and Threema," *Forensic Science International: Digital Investigation*, vol. 40, art. 301347, 2022.

[13] D. Sudiana, C. H. Nuruddin, M. Rizkinia, and D. Husna, "Forensic analysis of WhatsApp disappearing message on unrooted Android using mobile device forensics methodology NIST SP 800-101r1," *Evergreen*, vol. 11, no. 1, pp. 516–524, 2024, doi: 10.5109/7172316.

[14] A. Vasilaras, D. Dosis, M. Kotsis, and P. Rizomiliotis, "Retrieving deleted records from Telegram," *Forensic Science International: Digital Investigation*, vol. 43, art. 301447, 2022.

[15] S. Yoo, Y. Park, and D. Jeong, "Preserving evidence integrity in full file system extractions: Assessing hash inconsistencies and potential solutions," *Journal of Forensic Sciences*, early view, 2026, doi: 10.1111/1556-4029.70362.

[16] European Parliament and Council, "Directive (EU) 2016/680 of 27 April 2016 on the protection of natural persons with regard to the processing of personal data by competent authorities for the purposes of the prevention, investigation, detection or prosecution of criminal offences or the execution of criminal penalties, and on the free movement of such data," *Official Journal of the European Union*, vol. L 119, pp. 89–131, May 2016.

[17] R. Ayers, S. Brothers, and W. Jansen, "Guidelines on mobile device forensics," NIST Special Publication 800-101 Revision 1, National Institute of Standards and Technology, May 2014, doi: 10.6028/NIST.SP.800-101r1.

[18] WhatsApp LLC, "WhatsApp encryption overview: Technical white paper," WhatsApp, 2025. [Online]. Available: <https://www.whatsapp.com/security/WhatsApp-Security-Whitepaper.pdf>

[19] Telegram, "MTProto mobile protocol," Telegram API Documentation. [Online]. Available: <https://core.telegram.org/mtproto>

[20] Android Open Source Project, "File-based encryption," Android Documentation. [Online]. Available: <https://source.android.com/docs/security/features/encryption/file-based>

[21] M. N. K. Saunders, P. Lewis, and A. Thornhill, *Research Methods for Business Students*, 9th ed. Harlow, U.K.: Pearson Education, 2023.

[22] A. R. Hevner, S. T. March, J. Park, and S. Ram, "Design science in information systems research," *MIS Quarterly*, vol. 28, no. 1, pp. 75–105, Mar. 2004, doi: 10.2307/25148625.

[23] B. J. Oates, M. Griffiths, and R. McLean, *Researching Information Systems and Computing*, 2nd ed. London, U.K.: SAGE, 2022.

[24] Android Developers, "Behavior changes: Apps targeting Android 12," Google. [Online]. Available: <https://developer.android.com/about/versions/12/behavior-changes-12>

[25] Android Developers, "Back up user data with Auto Backup," Google. [Online]. Available: <https://developer.android.com/identity/data/autobackup>

- [26] Google, "Target API level requirements for Google Play apps," Play Console Help. [Online]. Available: <https://support.google.com/googleplay/android-developer/answer/11926878>
- [27] Samsung, "Data protection," Samsung Knox Documentation: Samsung Knox Mobile Security White Paper. [Online]. Available: <https://docs.samsungknox.com/admin/fundamentals/whitepaper/samsung-knox-mobile-security/system-security/data-protection/>
- [28] Samsung, "Samsung Auto Blocker," Samsung Knox Documentation: Samsung Knox Mobile Security White Paper. [Online]. Available: <https://docs.samsungknox.com/admin/fundamentals/whitepaper/samsung-knox-mobile-security/system-security/samsung-auto-blocker/>
- [29] P. Lorentz and H. Mahalik, "Android data collection simplified," Cellebrite, Jul. 20, 2020. [Online]. Available: <https://cellebrite.com/en/blog/android-data-collection-simplified/>
- [30] Cellebrite, "Android Backup APK Downgrade Extraction," Cellebrite Glossary. [Online]. Available: <https://cellebrite.com/en/glossary/android-backup-apk-downgrade-extraction-mobile-device-forensics/>
- [31] Privacy International, "A technical look at phone extraction," Oct. 14, 2019. [Online]. Available: <https://privacyinternational.org/long-read/3256/technical-look-phone-extraction>
- [32] J. Dowrick, P. Modesti, and B. Findlay, "Android anti-forensics: A systematic review of applications, techniques, and investigative challenges," *Forensic Science International: Digital Investigation*, vol. 58, art. 302159, 2026, doi: 10.1016/j.fsidi.2026.302159.
- [33] European Parliament and Council, "Regulation (EU) 2016/679 (General Data Protection Regulation)," *Official Journal of the European Union*, vol. L 119, pp. 1–88, May 2016.
- [34] B. Garamvölgyi, K. Ligeti, A. Ondrejová, and M. von Galen, "Admissibility of evidence in criminal proceedings in the EU," *eucri*, no. 3, pp. 201–208, 2020. [Online]. Available: <https://eucri.eu/articles/admissibility-evidence-criminal-proceedings-eu/>
- [35] L. Bachmaier, "Mutual admissibility of evidence and electronic evidence in the EU," *eucri*, no. 2, pp. 223–229, 2023. [Online]. Available: <https://eucri.eu/articles/mutual-admissibility-of-evidence-and-electronic-evidence-in-the-eu/>
- [36] European Network of Forensic Science Institutes, "Best practice manual for the forensic examination of digital technology," ENFSI-BPM-FIT-01, ver. 01, Nov. 2015. [Online]. Available: https://enfsi.eu/wp-content/uploads/2016/09/1._forensic_examination_of_digital_technology_0.pdf
- [37] European Parliament and Council, "Regulation (EU) 2023/1543 on European Production Orders and European Preservation Orders for electronic evidence in criminal proceedings and for the execution of custodial sentences following criminal proceedings," *Official Journal of the European Union*, vol. L 191, pp. 118–180, Jul. 2023.

Appendices

Appendix A: Extraction logs and screenshots

The screens below were captured during acquisition attempts A1 to A4 (Table 4.1). IMEI, Android ID and Advertising ID have been blacked out. Every other capture is held with the extraction files on encrypted storage.

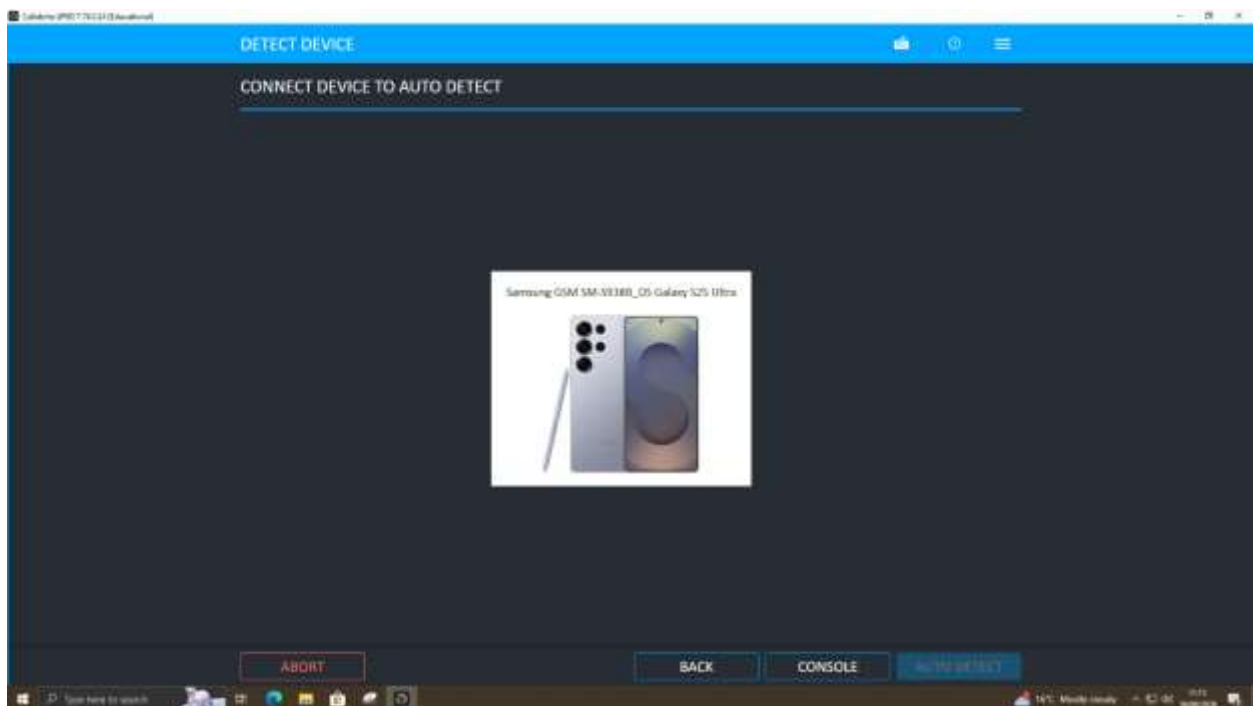


Figure A.1: UFED detects the handset as Samsung SM-S938B_DS Galaxy S25 Ultra (16/09/2026, 11:15)

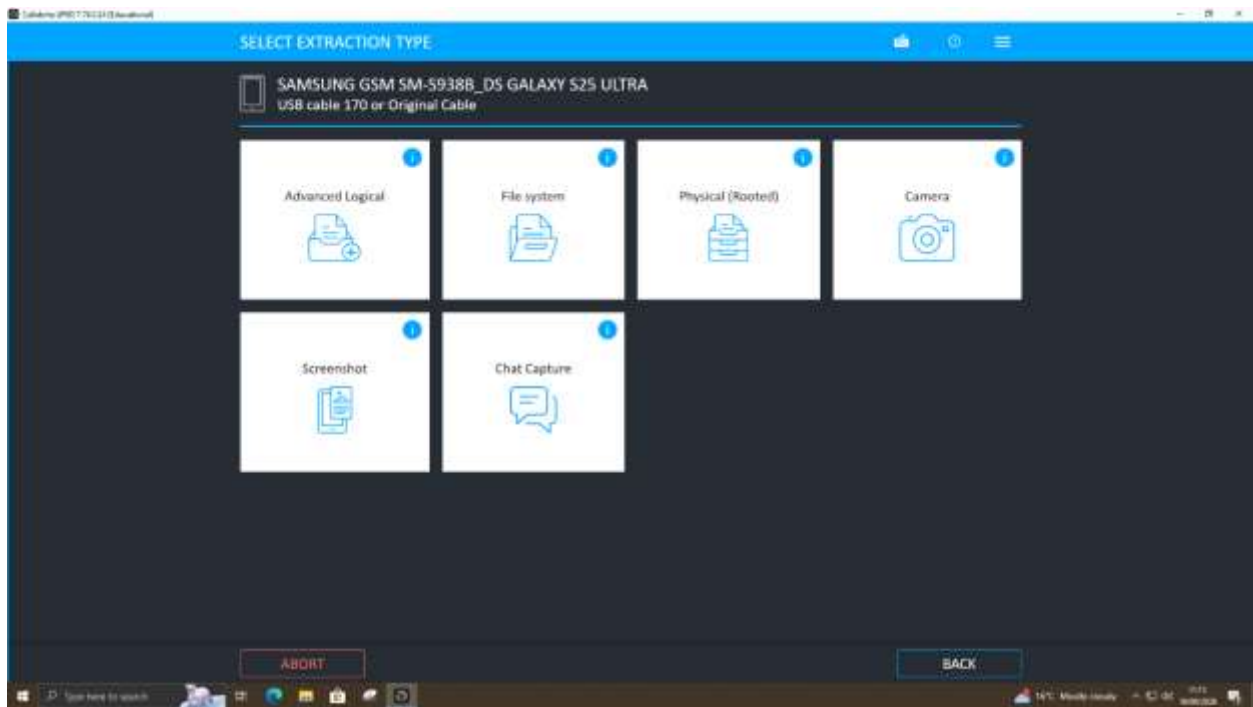


Figure A.2: Extraction types offered by UFED 4PC 7.76.0.24 for the SM-S938B

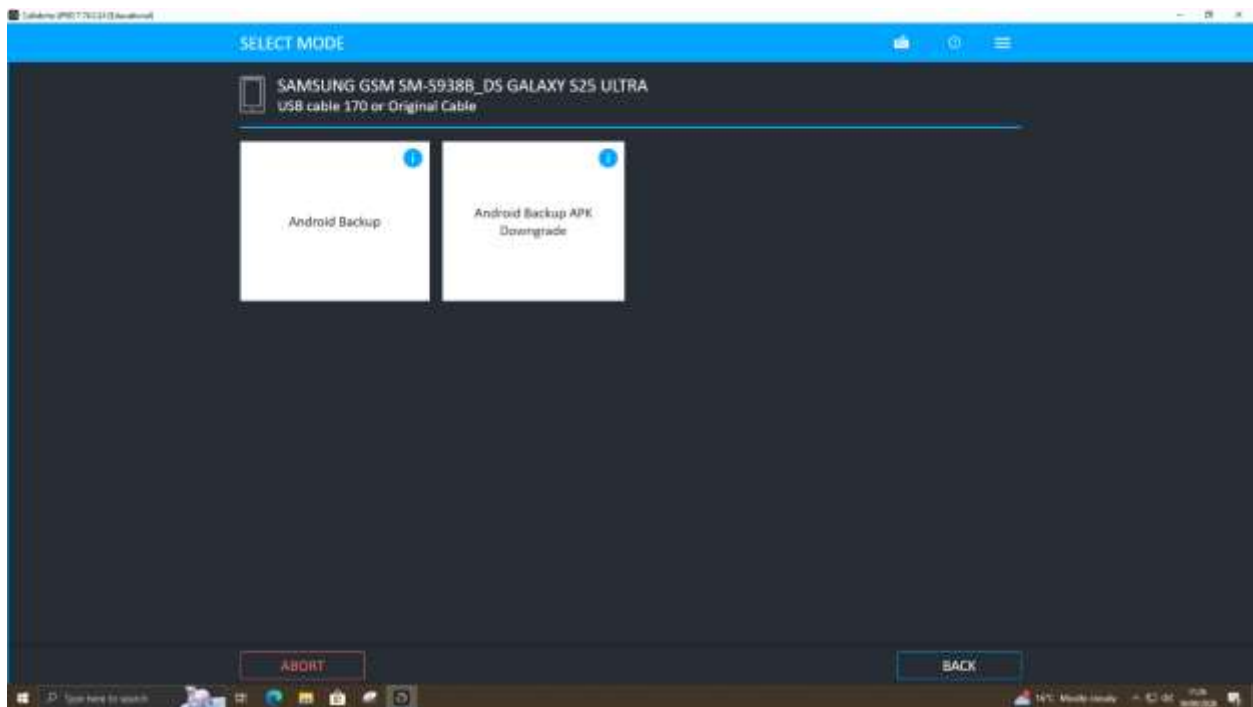


Figure A.3: File System modes offered: Android Backup and Android Backup APK Downgrade

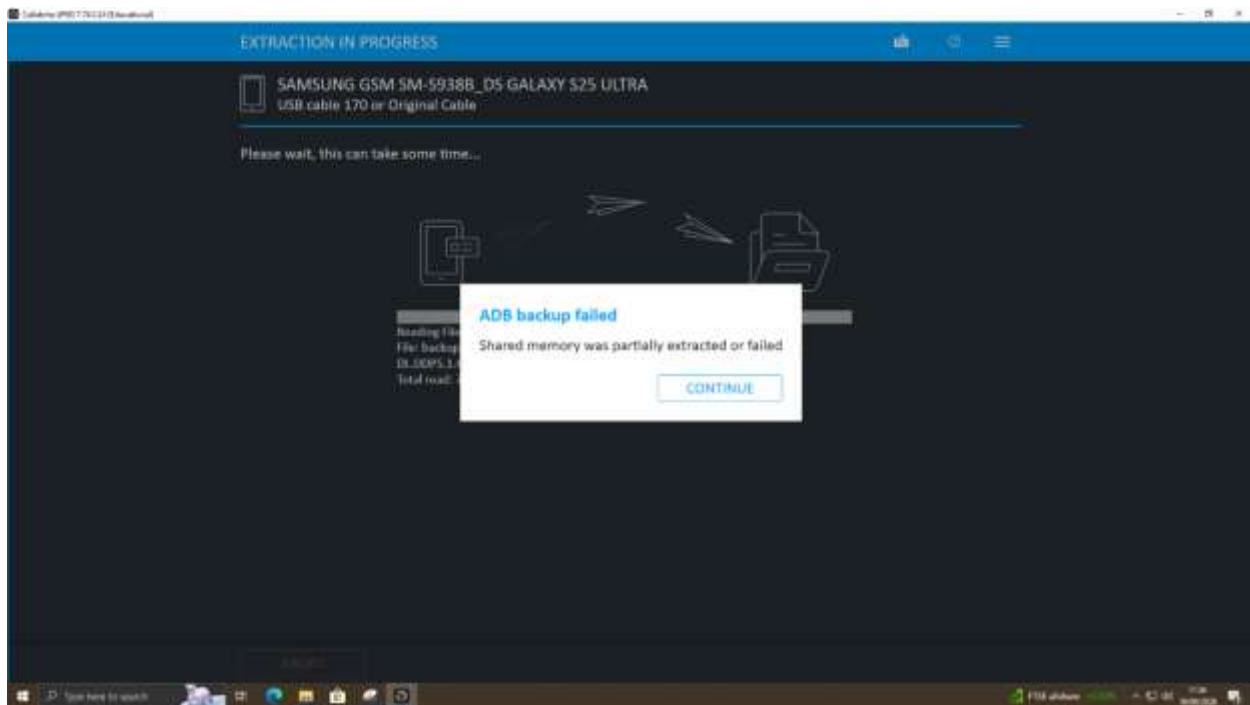


Figure A.4: Attempt A1 fails with "ADB backup failed. Shared memory was partially extracted or failed" (16/09/2026, 11:36)

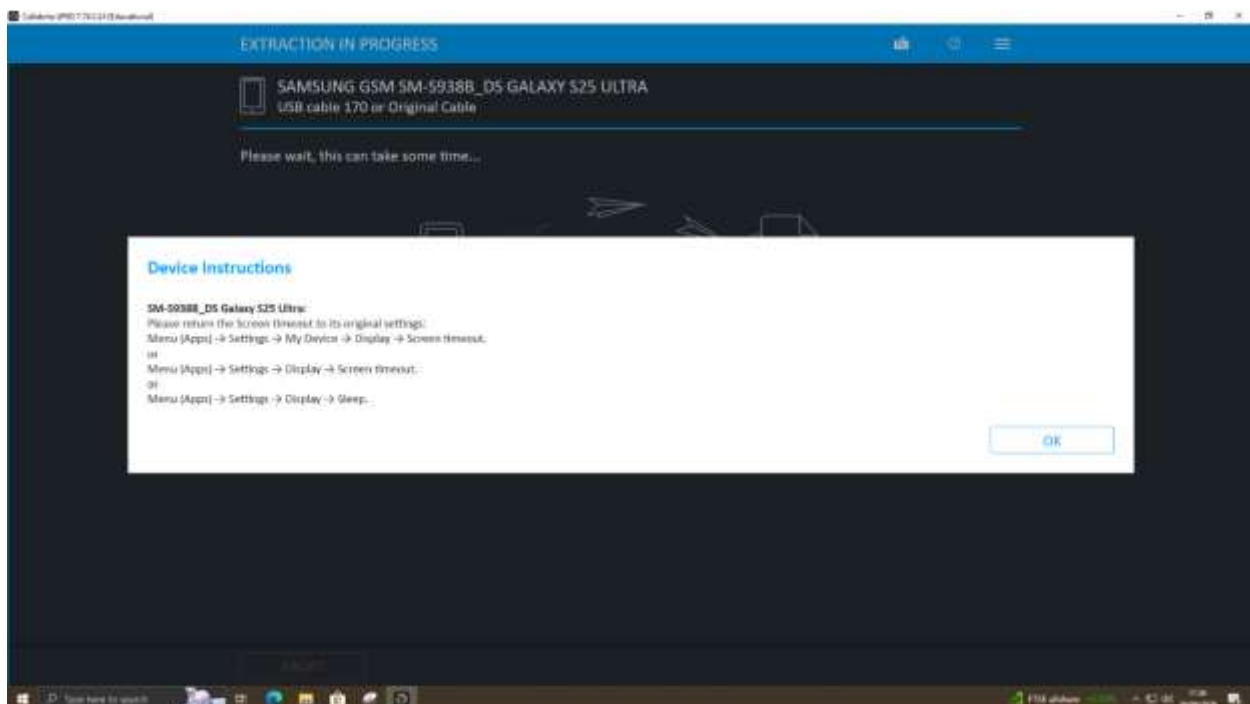


Figure A.5: Device instruction displayed after A1, asking for the screen timeout to be restored

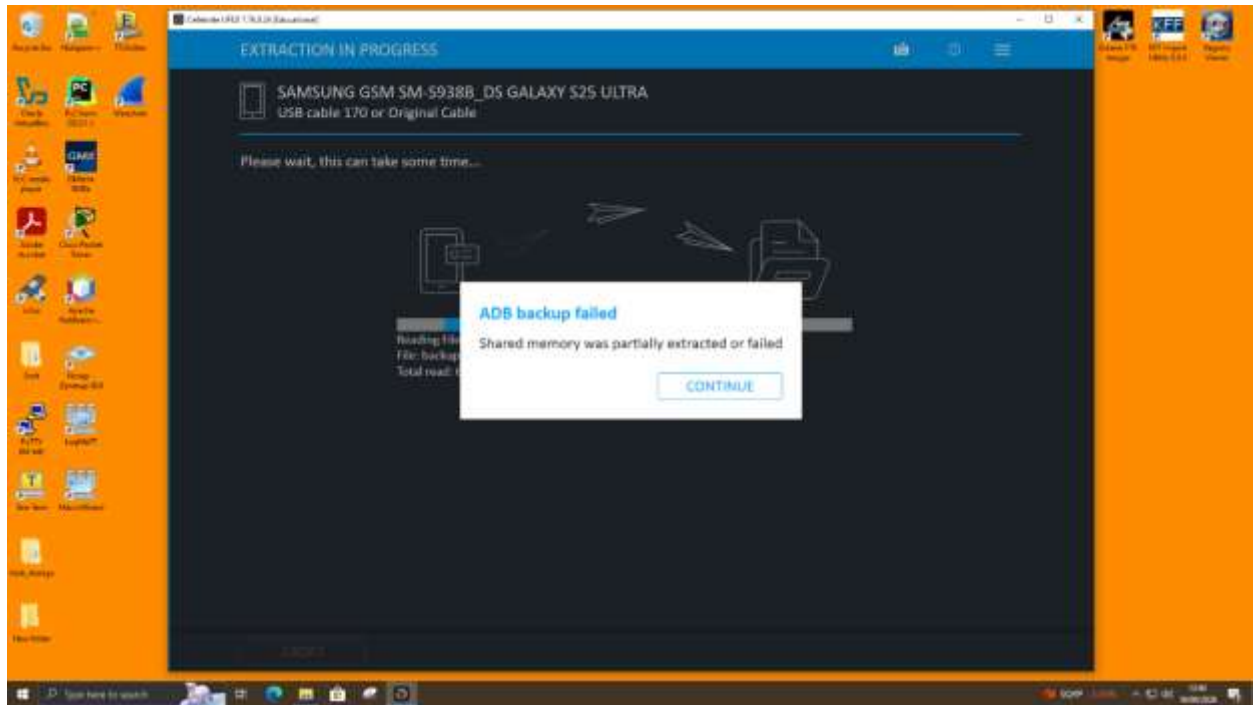


Figure A.6: Attempt A2 fails with the same error (16/09/2026, 12:40)

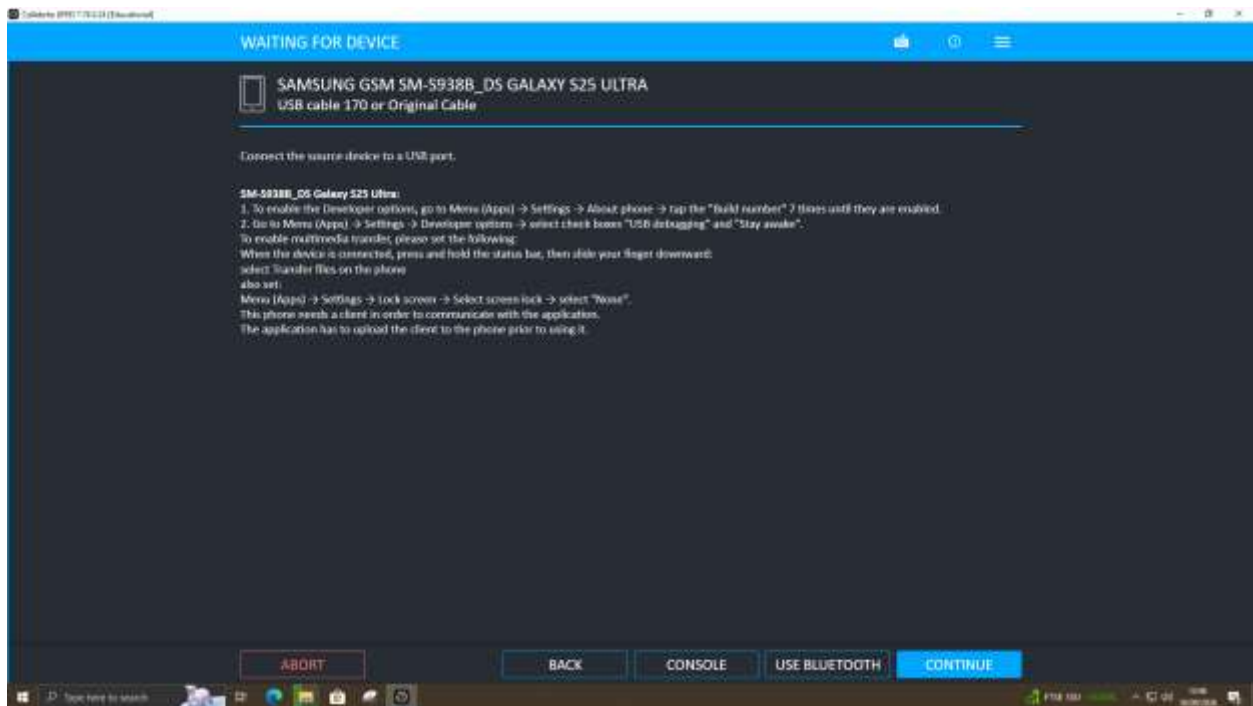


Figure A.7: UFED connection instructions for the SM-S938B before A3

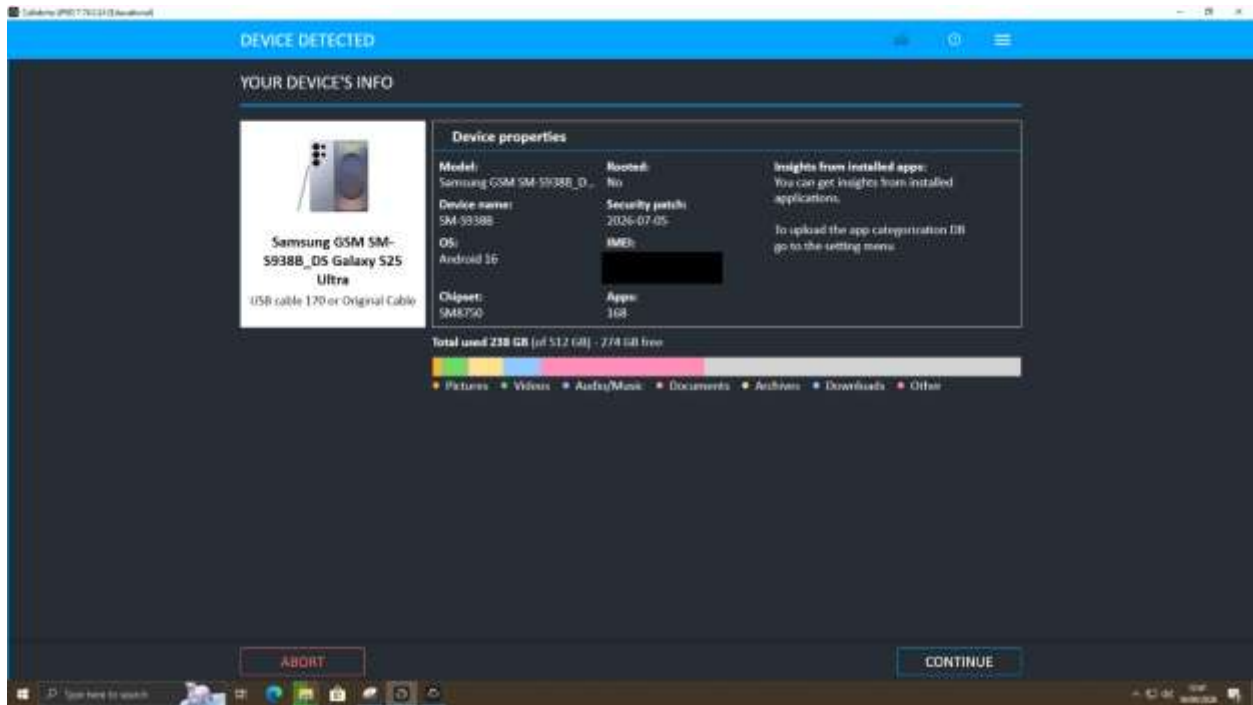


Figure A.8: Device properties read at A3: not rooted, Android 16, security patch 2026-07-05, 168 applications (IMEI redacted)

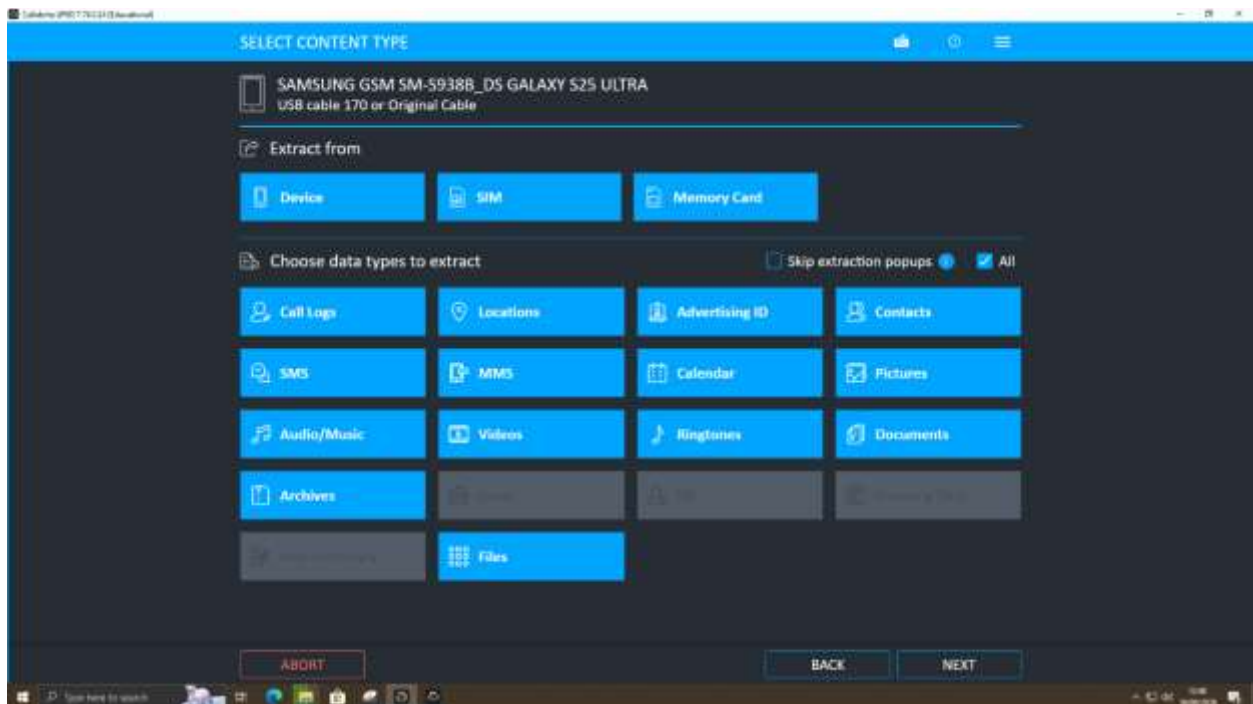


Figure A.9: Content selection at A3; Email, IM, Browsing Data and User Dictionary unavailable

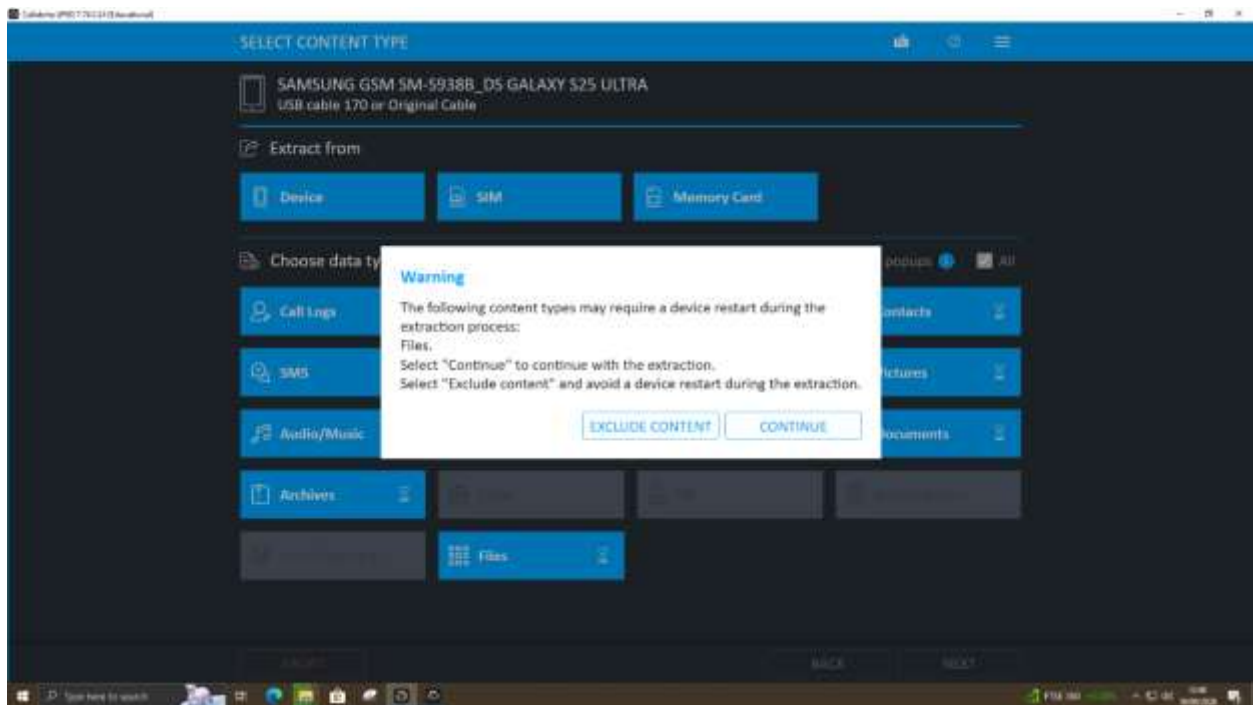


Figure A.10: UFED warning that the Files content type may require a device restart

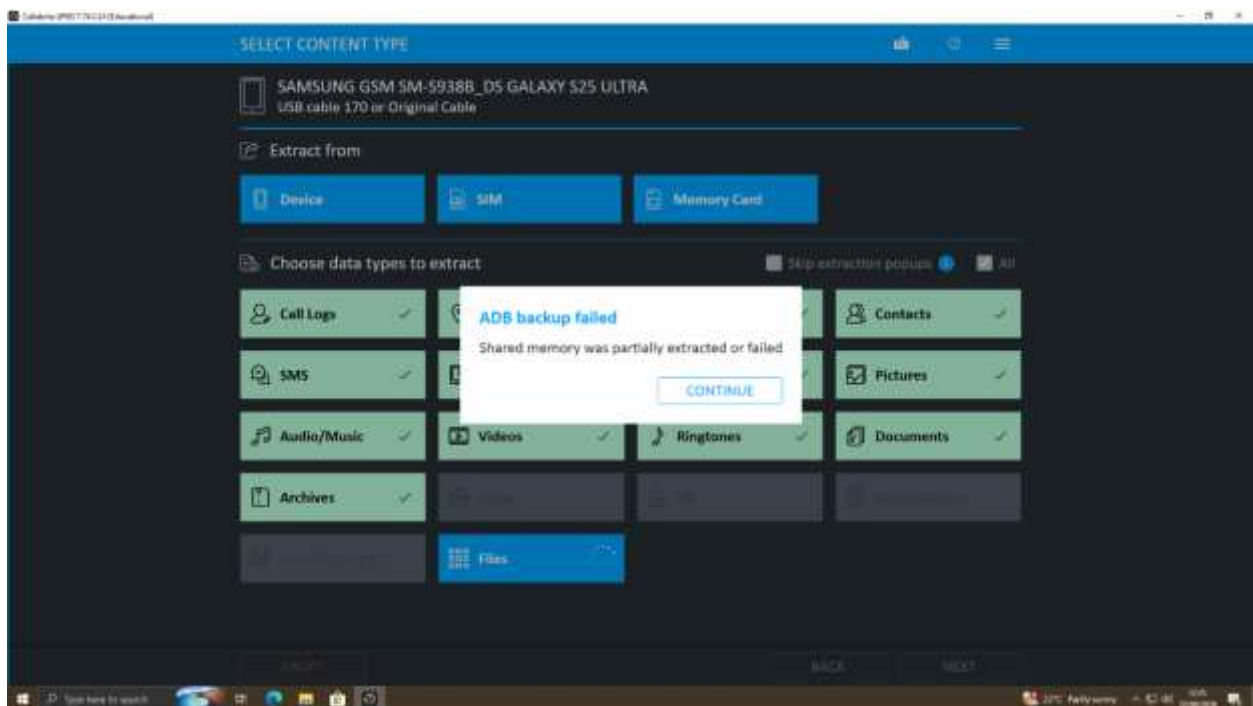


Figure A.11: Attempt A4 raises "ADB backup failed" during the Files stage (22/09/2026)

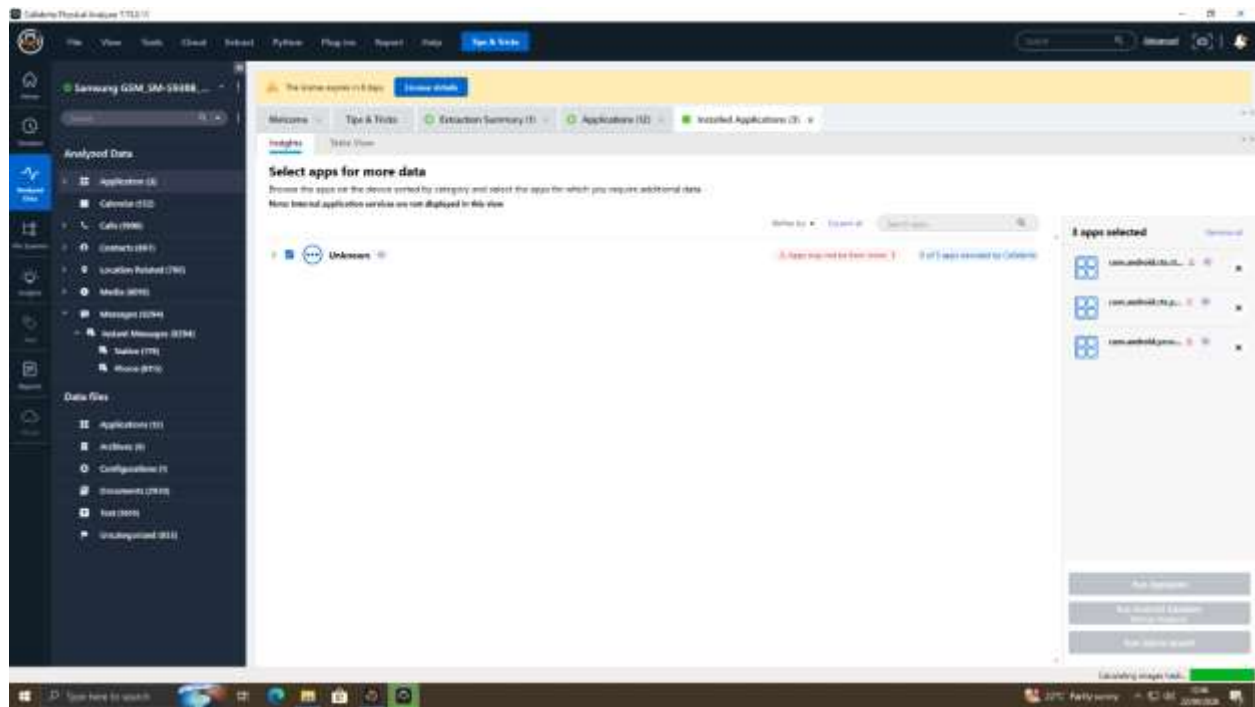


Figure A.14: Physical Analyzer application selection for A4: three system packages listed, 0 of 3 decoded; WhatsApp, Signal and Telegram absent

Appendix B: Hash values and chain-of-custody records

UFED wrote a SHA-256 value for each completed extraction at the point of acquisition, and Table B.1 lists them. Table B.2 is the custody log for the handset and the extraction files across the testing period. A1 and A2 left no extraction file, so there is nothing for them to hash.

Table B.1: SHA-256 values recorded by UFED at acquisition

Attempt	Date	File	SHA-256
A3	16/09/2026	Samsung GSM_SM-S938B_DS Galaxy S25 Ultra.zip	A853A2D9D3E9BC4E83E721E16E6EF53B012BE6F0087DFB31B3651B5AFB1B1D70
A3	16/09/2026	SummaryReport.pdf	EF784ACEEF61B8F6C35DA7A98594ACAF8BAF4B9D484AB4959619E5246A461DE
A4	22/09/2026	Samsung GSM_SM-S938B_DS Galaxy S25 Ultra.zip	EF0083D0CA866DE3D6D00C7CDAC51094A0F212DA41836B091E5B3E98435FBF3E
A4	22/09/2026	SummaryReport.pdf	337FE871774A351D02195433860E11D69D0AC82CBFED699900C7D9D79375515F

Table B.2: Chain-of-custody log

Date	Time	Event	Custodian	Record
06–07/09/2026	-	Corpus of 390 items populated on the handset	Researcher	Ground truth log
11/09/2026	11:20–11:28	Device state and application versions recorded	Researcher	Figures C.1–C.5
16/09/2026	11:11	A1 case opened; handset in airplane mode	Researcher	Figure A.1
16/09/2026	11:26–11:36	A1 File System (Android Backup) extraction; failed, no output	Researcher	Figure A.4
16/09/2026	11:42	A2 case opened	Researcher	Screenshot record
16/09/2026	12:01–12:40	A2 File System (Android Backup) extraction; failed, no output	Researcher	Figure A.6
16/09/2026	12:45	A3 case opened	Researcher	Screenshot record
16/09/2026	13:31:06–14:00:07	A3 Advanced Logical extraction;	Researcher	Table B.1

		partially completed; output hashed		
16–22/09/2026	-	Handset out of airplane mode between sessions; firmware update received (patch 2026-07-05 to 2026-08-05)	Researcher	Section 4.1
22/09/2026	11:38:42–12:25:24	A4 Advanced Logical extraction; partially completed; output hashed	Researcher	Figures A.11–A.12; Table B.1
22/09/2026	12:29:33	A4 loaded into Physical Analyzer 7.73.0.11; decoding started	Researcher	Figure A.13
From 22/09/2026	-	Extraction files held on AES-256 encrypted external storage	Researcher	Section 3.6

Appendix C: Device and app configuration details

The handset and applications were recorded on 11 September 2026, before the first acquisition. Table C.1 sets those values beside what UFED read during A3 and A4. The serial number, telephone numbers and IMEIs are blacked out.

Table C.1: Device and application configuration

Property	11/09/2026 (device settings)	A3, 16/09/2026 (UFED)	A4, 22/09/2026 (UFED / Physical Analyzer)
Model	SM-S938B/DS Galaxy S25 Ultra	SM-S938B DS	SM-S938B
Android version	16	16	16
One UI version	8.5	-	-
Build	BP4A.251205.006.S938BXXSBCZG3	-	BP4A.251205.006.S938BXXSCCZH1
Security patch level	-	2026-07-05	2026-08-05
Knox version	3.13	-	-
Rooted	-	No	No
Live encryption state	-	-	File-based
Installed applications	-	168	0 captured
Storage used	-	238 GB of 512 GB	-
Signal	8.26.4	-	-
Telegram	12.10.1	-	-
WhatsApp	2.26.35.75	-	-

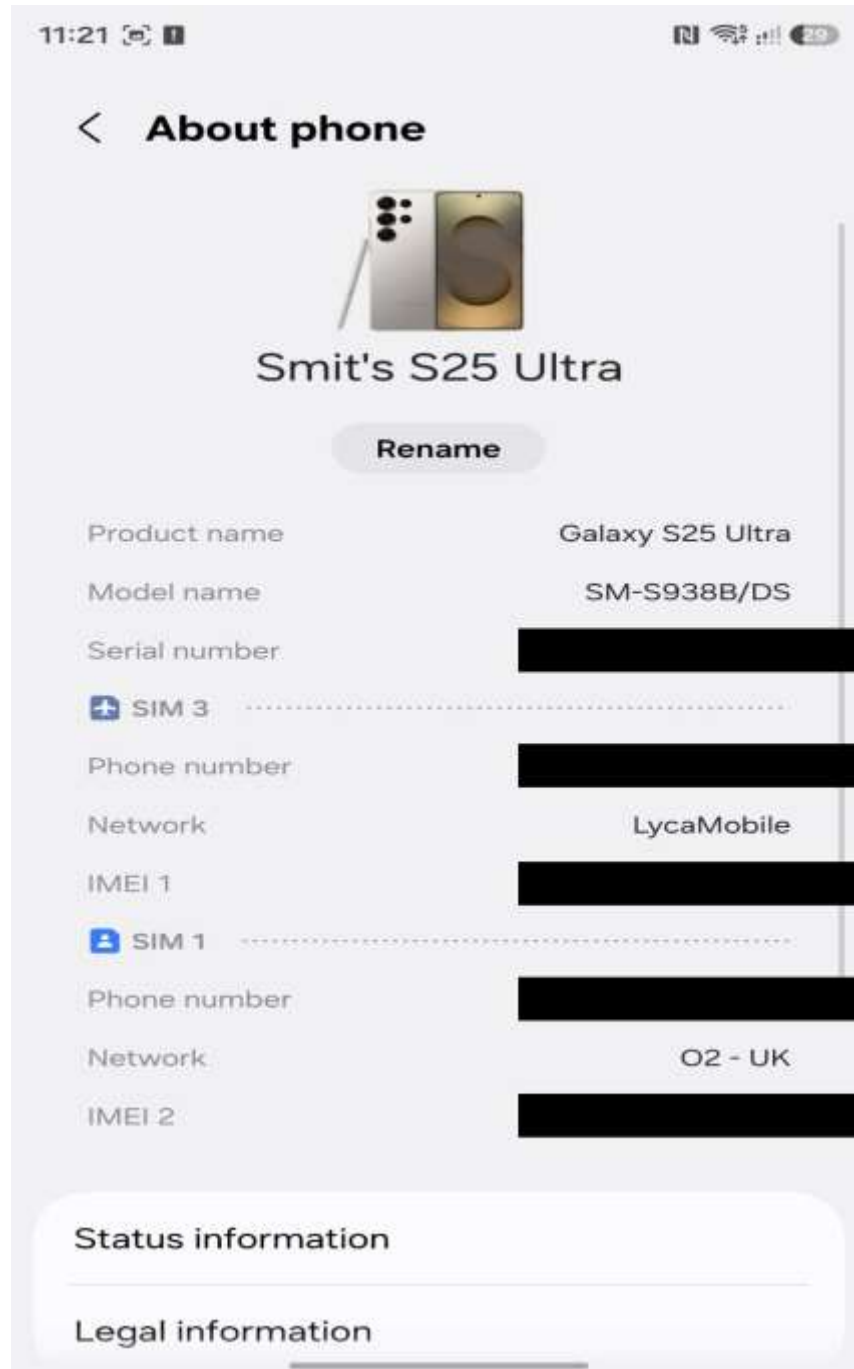


Figure C.1: About phone, 11/09/2026 (serial number, telephone numbers and IMEIs redacted)



Figure C.2: Software information: One UI 8.5, Android 16, build S938BXXSBCZG3, Knox 3.13

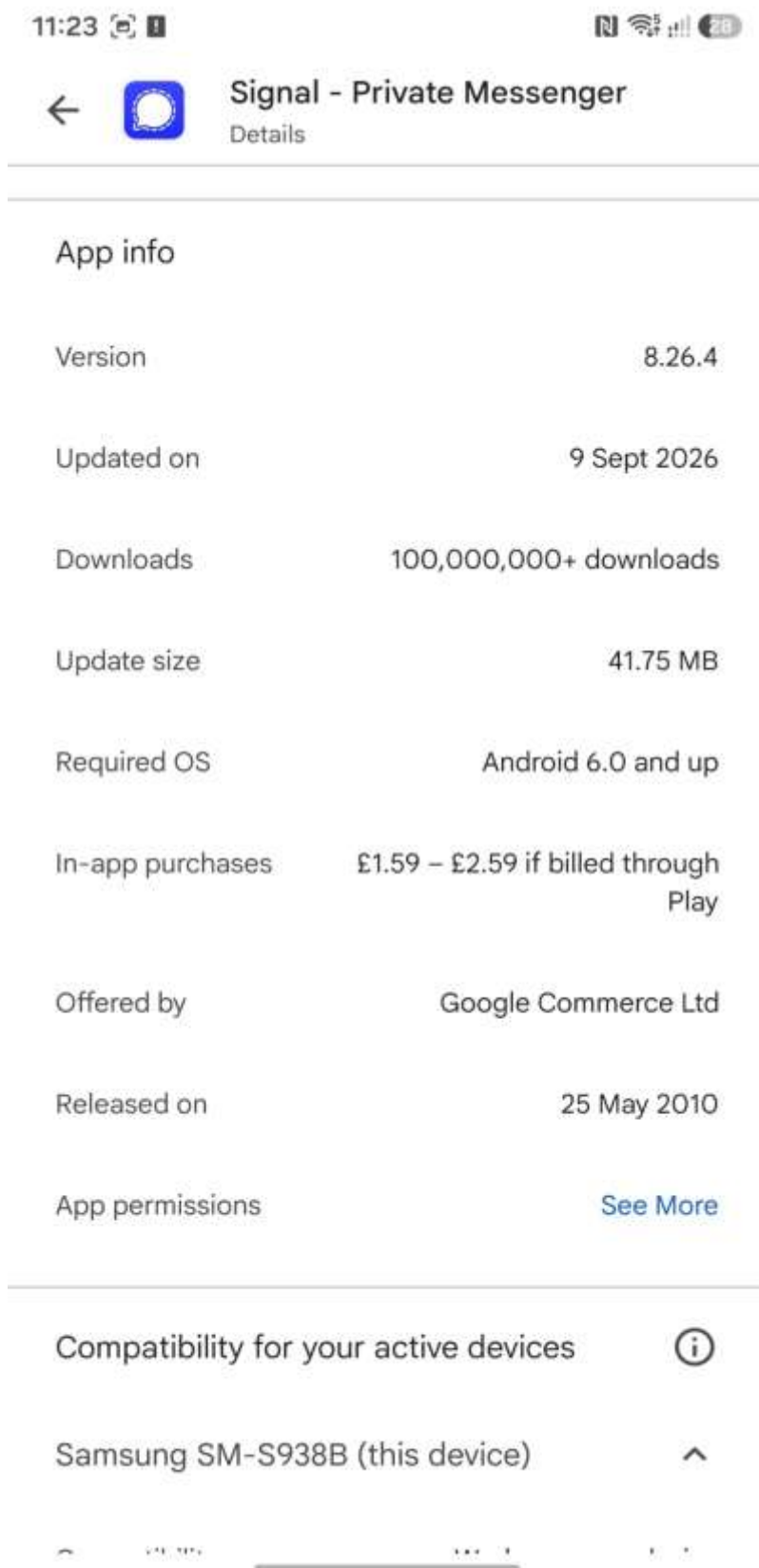


Figure C.3: Signal 8.26.4, Google Play app information

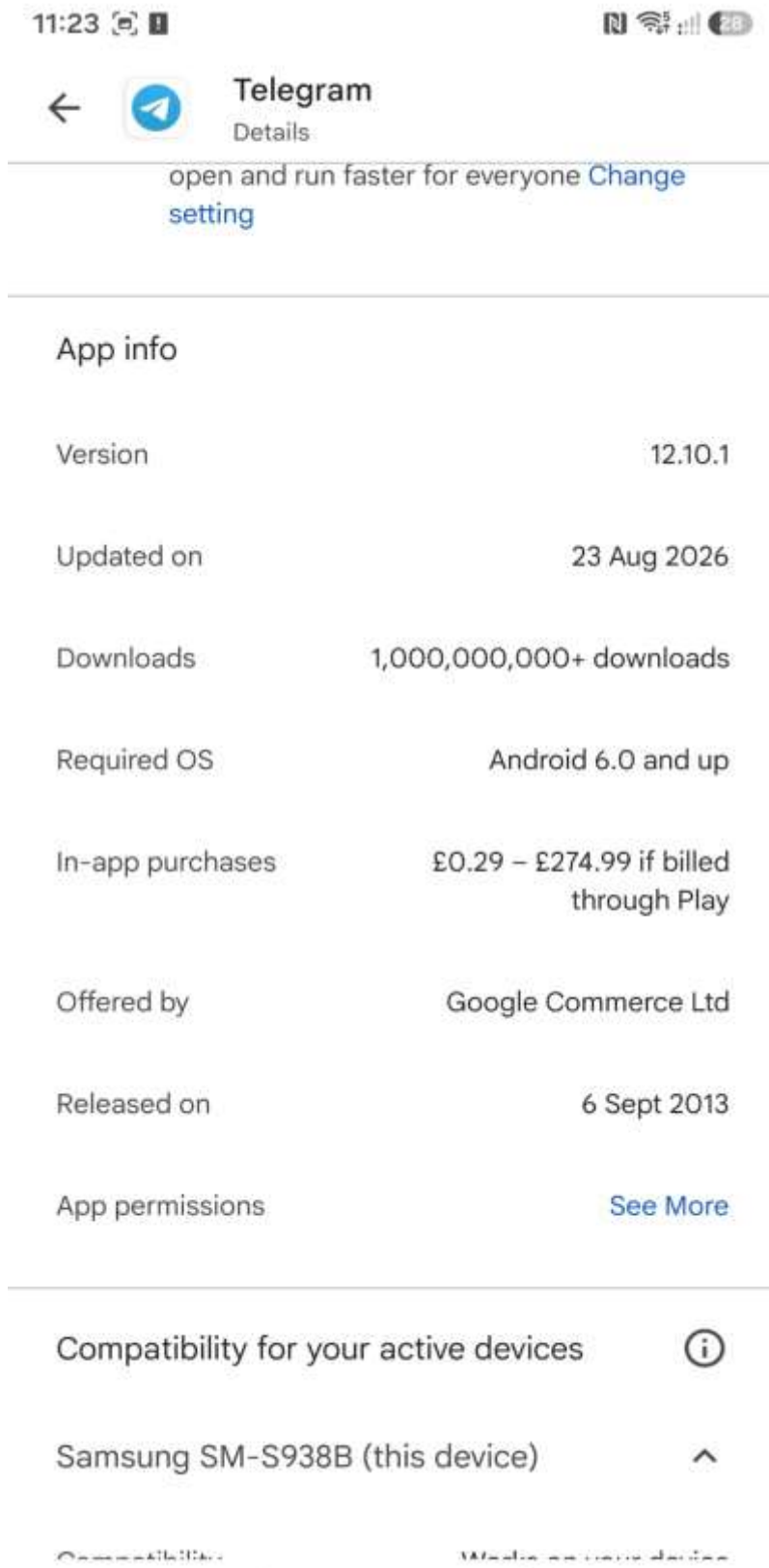


Figure C.4: Telegram 12.10.1, Google Play app information

11:23



WhatsApp Messenger

Details

12+

Rated for 12+

Parental guidance recommended

Users interact, Shares location

[Learn more](#)



Wear OS

Enhanced by the developer for Wear OS

App info

Version

2.26.35.75

Updated on

8 Sept 2026

Downloads

10,000,000,000+ downloads

Required OS

Android 6.0 and up

In-app purchases

₹25.00 – ₹22,400.00 per item

Offered by

WhatsApp LLC

Released on

18 Oct 2010

App permissions

[See More](#)

Figure C.5: WhatsApp 2.26.35.75, Google Play app information

Appendix D: Ethics approval documentation

The approved ethics application (ACFA7004), signed by the applicant and the supervisor on 5 August 2026, is submitted separately through the university's ethics submission link and is not reproduced here.