COMPUTER FORENSICS LAB

London digital forensics & e-discovery · Established 2007

Android Deleted Data, SQLite and Unallocated Space

Where Deleted Android Data Survives, Why Depth Decides It, and How Encryption and TRIM Change the Odds

On Android, as on iOS, most data lives in SQLite databases that mark deleted records as free rather than wiping them, so deleted messages, calls and app records frequently linger in free space and journals, recoverable until overwritten. But Android's deleted-data picture has its own shape. The depth of acquisition decides everything: a logical extraction sees no deleted data at all, a file-system extraction reaches the database remnants, and only a physical extraction reaches the unallocated space where deleted files survive. And two modern realities cut against recovery: file-based encryption means deleted data in unallocated space is unreadable without the keys, and flash storage's TRIM function may erase deleted blocks quickly, so the classic "carve the free space" recovery works far less often on modern devices than it once did. For the lawyer, this makes deleted-data recovery on Android a question of what depth was reached, what the device's encryption and storage permit, and how promptly it was preserved, with the estate as the route around what the device no longer holds. This guide sets out for UK lawyers where deleted Android data survives, how depth, encryption and TRIM govern recovery, and how deleted data is recovered and read honestly.

⏱ Estimated reading time: 24 min read

§ ABOUT THE AUTHOR

PREPARED BY COMPUTER FORENSICS LAB E-DISCOVERY TEAM

Computer Forensics Lab is a London-based digital forensics and electronic disclosure practice established in 2007. Android deleted-data recovery is a core technical strength of its mobile work: recovering deleted records from SQLite free space and journals, carving unallocated space where the device permits, understanding how file-based encryption and TRIM limit modern recovery, and turning to backups and the estate where the device no longer holds the data. The analysis states honestly what was recovered and what could not be, and is reported under CPR Part 35 and CrimPR Part 19.

ESTABLISHED 2007 · LONDON | ISO 17025-ALIGNED PROCEDURES | SQLITE & UNALLOCATED-SPACE RECOVERY | ENCRYPTION & TRIM AWARE | CPR PART 35 EXPERT REPORTS | FULL CHAIN-OF-CUSTODY DOCUMENTATION

§ CONTENTS

In this guide

§ 0 1 · ORIENTATION

Executive summary

The headline point: deleted Android data frequently survives in SQLite free space and journals, recoverable until overwritten, but the depth of acquisition decides what can be reached, and file-based encryption and flash TRIM mean that classic unallocated-space carving works far less often on modern devices, so recovery depends on depth, the device's encryption and storage, and prompt preservation, with the estate as the route around what the device no longer holds.

Where it survives (§3): in SQLite free space, write-ahead logs and journals, in app trash and recently-deleted areas, in unallocated space, and in backups and the estate. **Depth decides (§4):** logical sees no deleted data, file-system reaches database remnants, physical alone reaches unallocated space. **Encryption and TRIM (§5):** file-based encryption makes unallocated data unreadable without keys, and TRIM may erase deleted blocks quickly, cutting the odds on modern devices. **Recovery (§6):** deleted records recovered and reassembled from the reachable areas, completeness stated. **Deployment (§7):** restoring what a party erased, from the device where it permits and from backups and the estate where it does not.

- **Deleted is not gone:** SQLite marks records free; remnants linger until overwritten.
- **Depth decides:** logical sees none; file-system reaches remnants; physical reaches unallocated.
- **Encryption and TRIM cut the odds:** modern devices carve far less often.
- **Preserve promptly:** use overwrites remnants; TRIM erases blocks.
- **The estate is the route around:** backups, the Google account and counterparts hold what the device lost.

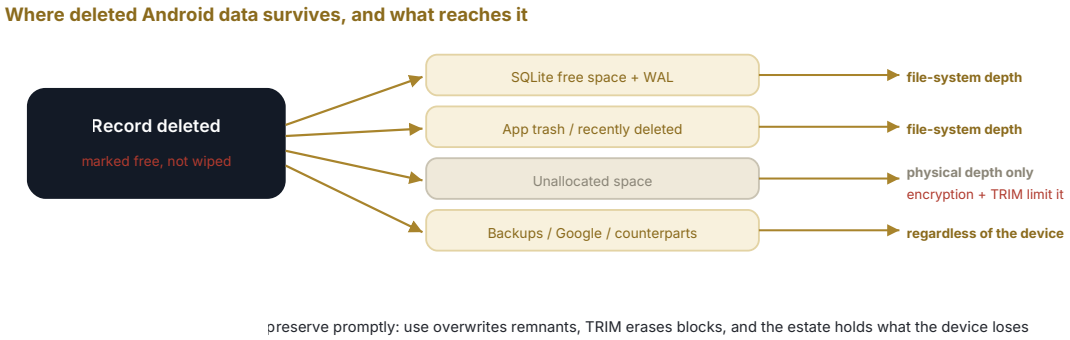

Figure 1 - deleted Android data survives in SQLite free space, app trash, unallocated space and the estate, but each is reached only at a given depth, and encryption and TRIM limit what unallocated space yields on modern devices.

§ 0 2 · FIRST PRINCIPLES

The problem in plain English: deleted is not gone, but the odds have changed

The iPhone deleted-data guide (guide 155) explained the key to recovery on any modern phone: almost everything lives in SQLite databases, and when a record is deleted SQLite usually does not erase it but marks its space as free, so the content lingers in the database's own free space and in its write-ahead log and journals until something overwrites it. All of that is true of Android, whose apps and system store their data in SQLite just as iOS does. So on Android, too, a deleted message, call or app record very often survives inside the very database it was deleted from, recoverable by skilled analysis, provided the phone is preserved before continued use reuses the space. That is the encouraging half of the picture, and it is the half that most often decides a case.

The other half is that Android's deleted-data picture has its own shape, and in two respects the odds have changed against the classic techniques. The first is depth. Android's three depths of acquisition (guide 162) reach very different things: a logical extraction sees no deleted data at all, a file-system extraction reaches the databases and their remnants, and only a physical extraction reaches the unallocated space of the storage, where deleted files and fragments survive outside any database. So what deleted data can be recovered is decided before analysis begins, by the depth the device permitted. The second is the modern hardware. Older recovery relied heavily on carving unallocated space for deleted files, and on modern Android devices that works far less often. File-based encryption (guide 163) means that data in unallocated space is encrypted with keys that may no longer exist once the file is deleted, so even a physical extraction may yield ciphertext that cannot be read. And flash storage's TRIM function, which tells the storage controller that deleted blocks can be erased, may cause deleted data to be physically cleared soon after deletion, so there may be nothing left to carve. The upshot is that deleted-data recovery on Android is above all a question of what depth was reached, what the device's encryption and storage permit, and how promptly the phone was preserved, and that where the device no longer holds the data, the estate, backups, the Google account, the counterparts, is the route around it. This guide sets out where deleted Android data survives, how depth, encryption and TRIM govern it, and how it is recovered and read honestly.

§ 0 3 · WHERE IT SURVIVES

Where deleted Android data survives

- **SQLite free space:** deleted records lingering in the free pages of app and system databases, their content often intact until the space is reused, the primary remnant on Android as on iOS (guide 155): the database remnant.
- **Write-ahead logs and journals:** recent deletions and prior versions held in the databases' WAL and rollback journals, so a record deleted shortly before acquisition frequently survives there (guide 155): the journal copy.
- **App trash and recently-deleted areas:** gallery, file-manager and app trash folders and recently-deleted features holding deleted items intact for a period (guide 166): the grace-period copy.
- **Unallocated space:** deleted files and fragments surviving in the storage's unallocated space outside any database, reachable only by physical extraction and subject to encryption and TRIM (§4, §5): the carveable remnant, where it exists.
- **Backups and the estate:** earlier states in local, manufacturer and Google backups, and copies on counterpart devices and in the cloud (guides 174, 164), holding data the device has lost: the preserved prior state.

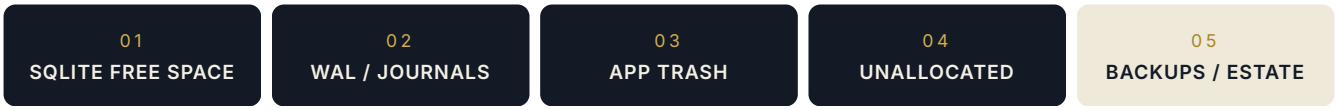

Figure 2 - the places deleted Android data survives, from SQLite remnants and journals through app trash and unallocated space to backups and the estate.

§ 0 4 · DEPTH DECIDES

Depth decides: what each extraction reaches

- **Logical: no deleted data:** a logical extraction taking what the device hands over, which never includes deleted records or database internals, so it recovers nothing deleted and its silence proves nothing (guide 162): the blind depth.
- **File system: the database remnants:** a file-system extraction reaching the databases, their free space and journals, so deleted records within them are recoverable, the depth that decides most cases (guide 162): the remnant-reaching depth.
- **Physical: unallocated space too:** a physical extraction alone reaching unallocated space and deleted files outside databases, the most complete but increasingly rare on modern devices (guide 162): the complete depth.
- **Assess what the device permits:** the achievable depth assessed for this specific device before expectations of deleted-data recovery are set (guide 161): the device-decided reach.
- **Read every report by its depth:** a claim that no deleted data exists read against whether the method could have found any (guide 162): the depth check on every claim.

§ 0 5 · ENCRYPTION AND TRIM

Encryption and TRIM: the modern limits

- **File-based encryption:** each file encrypted with its own key on modern devices, so once a file is deleted and its key discarded, its remnants in unallocated space are unreadable ciphertext even to a physical extraction (guide 163): the encryption limit.
- **TRIM on flash storage:** the storage controller told which deleted blocks it may erase, and often erasing them soon after, so deleted data may be physically gone from unallocated space within a short time (guide 4): the TRIM limit.
- **Database remnants largely survive these:** deleted records inside a live database's free space and journals sitting within a file that is still allocated and decrypted while the device is unlocked, so they survive encryption and TRIM in a way unallocated data does not (§3): the surviving remnant.
- **Older and unencrypted devices differ:** older devices, and unencrypted SD cards (guide 166), lacking these protections, so classic carving may still work on them: the device-specific odds.
- **State the effect honestly:** the effect of encryption and TRIM on what could be recovered stated plainly, so a failure to carve unallocated space is explained rather than left unexplained (guide 100): the honest modern limit.

§ 0 6 · RECOVERY AND HONEST LIMITS

Recovery, interpretation and honest limits

- **Recover from the reachable areas:** deleted records carved from database free space and journals at file-system depth, and from unallocated space where physical depth, encryption and TRIM permit (§3, §4): the recovery.
- **Reassemble and read correctly:** recovered fragments reassembled into coherent records with their fields and relationships, and read for what they truly are, since a misassembled fragment misleads (guide 155): the accurate reassembly.
- **Completeness stated:** whether each recovered record is whole or partial stated, so a fragment is never presented as a whole (guide 155): the honest completeness.
- **Overwriting and erasure limit recovery:** the central limits, that continued use reuses free space and TRIM erases blocks, so recovery is never guaranteed and prompt preservation is decisive (§2): the time-critical limit.
- **Attribution, dating and confidence:** recovered records attributed with care, dated where possible, any deletion assessed in context, and confidence stated honestly (guides 105, 110, 100): the honest recovery account.

§ 0 7 · DEPLOYMENT

Deployment: restoring what a party erased

- **Recovering deleted messages and calls:** deleted messages and call records recovered from the messaging and system databases' remnants, so an erased conversation or contact pattern is restored (guides 165, 153): the deleted communications restored.
- **Recovering deleted app records:** deleted records in third-party app databases recovered, so material removed from an app is restored (guide 167): the deleted app data restored.
- **Recovering deleted media:** deleted images recovered from trash, the media store index and, where carving works, unallocated space and SD cards (guide 166): the deleted media restored.
- **Dating a deletion:** recovered data and backup comparison dating when a deletion occurred, so the removal itself becomes evidence (guides 109, 174): the deletion dated.
- **Reaching what the device lost:** backups, the Google account and counterparts supplying deleted data the device's depth, encryption or TRIM prevented recovering (§8): the estate deployed.

§ 0 8 · THE WIDER MAP

Source architecture: where else the evidence lives

Per this series' source-architecture discipline: because depth, encryption and TRIM limit what the device itself yields, the estate matters more for deleted data on Android than on any earlier platform in this series. The phone holds database remnants at file-system depth and, where permitted and readable, unallocated remnants at physical depth. But local backups, the manufacturer's cloud and the Google account (guides 174, 164) hold earlier states of the databases with records since deleted, reachable regardless of the phone's lock; an unencrypted SD card (guide 166) holds its own remnants free of the phone's encryption and often carveable; the counterpart's device holds the other side of a deleted conversation; the carrier holds SMS and call records; and the app providers hold records the device no longer has. The discipline is to recover from the device's reachable areas first, and then, where depth, encryption or TRIM has put the data beyond the device, to turn to the pre-deletion backups, the cloud, the card and the counterparts, so that a deletion is defeated from a direction the device's limits cannot block. When the device permits only a logical extraction, the backups and account hold the remnants it cannot see; when encryption or TRIM has cleared unallocated space, a pre-deletion backup or the counterpart holds the record; and comparing the live database with a backup dates the deletion.

EVIDENCE	DEVICE (FILE-SYSTEM DEPTH)	DEVICE (PHYSICAL DEPTH)	BACKUPS / GOOGLE / MAKER CLOUD	SD CARD	COUNTERPART / CARRIER	RECOVERABLE?
Deleted messages	Y: SQLite remnants	Y: + unallocated (if readable)	Y: pre-deletion state	n/a	Y: their copy / carrier SMS	Often
Deleted calls	Y: remnants	Y	Y: if backed up	n/a	Y: carrier records	Often
Deleted app records	Y: app db remnants	Y	Varies (opt-in backup)	Some app data	App provider	Varies
Deleted files / media	Trash, index entries	Unallocated (encryption / TRIM permitting)	Y: Google Photos, backups	Y: carveable	Their copy	Varies
Overwritten / TRIMmed	Lost	Lost	Y: if pre-deletion	Maybe	Their copy	Via estate

Fallback strategy in one line: recover from the device's reachable remnants first; where depth, encryption or TRIM puts data beyond the device, pre-deletion backups, the Google account, the SD card and counterparts hold it, and a backup comparison dates the deletion.

§ 0 9 · IN THE WILD

Worked examples

EXAMPLE 1 · THE DELETED MESSAGES IN THE DATABASE'S FREE SPACE

A party deleted an incriminating conversation from their Android messaging app. The device §4 permitted a file-system extraction, and the §3 SQLite recovery restored the messages: they survived in the free space and write-ahead log of the app's database, and were §6 reassembled with participants, timing and content (guides 155, 165). Because the phone had been §2 preserved promptly, the free space had not been reused. The §6 completeness of each message was stated, and the §8 counterpart's device corroborated them. The deletion had removed the messages from view, not from the database. The lesson is §2's and §3's: on Android as on iOS, deleted records linger in SQLite free space and journals, and a file-system extraction reaches them, so a deleted conversation is very often recoverable, provided the device permits that depth and is preserved before continued use overwrites the remnants.

EXAMPLE 2 · THE UNALLOCATED SPACE THAT YIELDED ONLY CIPHERTEXT

A matter turned on deleted files, and the device permitted a §4 physical extraction, so a classic carve of unallocated space was attempted. It yielded little: the device used §5 file-based encryption, so the deleted files' remnants were ciphertext whose keys had been discarded on deletion, and §5 TRIM had already erased many of the deleted blocks. Rather than leave this unexplained, the §5 effect of encryption and TRIM was stated honestly, and the §8 estate pursued: the §8 Google account's backup and Google Photos held the files from before the deletion (guides 164, 166), and an §8 unencrypted SD card yielded carveable remnants. The device's unallocated space had failed, and the estate had not. The lesson is §5's: on modern Android devices file-based encryption and TRIM mean that carving unallocated space works far less often than it once did, so the failure is explained honestly and the pre-deletion backups, cloud and any unencrypted card are the routes to what the device's storage no longer readably holds.

EXAMPLE 3 · THE LOGICAL REPORT THAT COULD NEVER HAVE FOUND IT

A party's expert reported, on the basis of a §4 logical extraction, that no deleted data existed on a device. The §4 depth check exposed the claim: a logical extraction sees no deleted data at all, so its silence proved only that the method could not have found any (guide 162). The device permitted a file-system extraction, which was §7 sought and obtained, and the §3 database remnants yielded the deleted records the logical report had declared absent. The §6 report was corrected on its depth. The lesson is §4's: depth decides what deleted data can be reached, and a logical extraction reaches none, so any claim that no deleted data exists is read against the method that produced it, and a deeper extraction is sought where the device permits and the matter turns on deletion.

§ 10 · WHERE IT GOES WRONG

Common mistakes and technical limitations

Common mistakes

- **Accepting a logical report's "nothing deleted":** the cardinal error, when the method could never have found deleted data (Example 3, §4).
- **Expecting classic carving to work:** assuming unallocated space will yield files on a modern encrypted, TRIMmed device (Example 2, §5).
- **Not preserving promptly:** allowing use to overwrite database remnants and TRIM to erase blocks (§2, §6).
- **Ignoring the database remnants:** focusing on unallocated space when the SQLite free space and journals hold the records (Example 1, §3).
- **Presenting a fragment as whole:** a partial recovered record shown as complete (§6).
- **Ignoring the estate:** not turning to backups, the Google account, the SD card and counterparts when the device's limits block recovery (§8).
- **Not explaining a failed carve:** leaving unallocated-space failure unexplained rather than attributing it to encryption or TRIM (§5).
- **Ignoring the SD card:** missing an unencrypted card whose remnants are carveable (§5, §8).

Technical limitations

- **Depth decides reach:** logical sees no deleted data; the device sets the ceiling: the core limit (§4).
- **Encryption may render remnants unreadable:** deleted files' keys are discarded (§5).
- **TRIM may erase deleted blocks:** unallocated data can be physically gone quickly (§5).
- **Overwriting reuses free space:** database remnants are lost with continued use (§6).
- **Recovered data may be partial:** completeness is stated (§6).

§ 11 · INTERROGATORIES & DRAFTING AIDS

Questions to ask · Suggested wording

Ask your client

- Might relevant messages, calls, app records or files have been deleted, and has the phone been preserved promptly with its use minimised?
- What depth does the device permit, and does it use file-based encryption, so we know what deleted data is reachable?
- Are backups, the Google account, an SD card and the counterpart's device available, to reach what the device cannot?

Ask your opponent

- Please preserve the device without continued use that may overwrite or TRIM deleted data, and disclose the acquisition depth of any report you rely on and whether it could have reached deleted data.
- Please confirm whether relevant data has been deleted, and when, and preserve and disclose all backups, the Google account and any SD card.
- Where your report rests on a logical extraction, please agree to a deeper extraction or explain why the device does not permit one.

Ask your eDiscovery / forensic provider

- At the depth this device permits, what deleted data can be recovered, and from which areas?
- How do encryption and TRIM affect unallocated-space recovery on this device, and is a failed carve explained?
- What do backups, the Google account, the SD card and counterparts supply, and can the deletion be dated?

SUGGESTED WORDING · INSTRUCTION FOR AN ANDROID DELETED-DATA RECOVERY

> "We instruct you to recover deleted data from the Android device at Schedule 1 relevant to Schedule 2. Please, preserving the device from continued use and preserving integrity: (1) assess the acquisition depth this device permits and its encryption and storage characteristics, and advise what deleted data is reachable; (2) at file-system depth or deeper, recover deleted records from the SQLite databases' free space, write-ahead logs and journals, and from app trash and recently-deleted areas; (3) where physical depth is available, carve unallocated space, and state honestly the effect of file-based encryption and TRIM on what it yields; (4) recover earlier states from local, manufacturer and Google backups and any SD card, and compare them to date any deletion; (5) reassemble recovered records into coherent form, state whether each is complete or partial, attribute them with care and corroborate with counterparts and carrier records; and (6) state what was recovered, what could not be and why, and the confidence in it honestly, never reading a shallow extraction's silence as proof that deleted data does not exist."

§ 12 · QUICK CONTROL

Checklist and red flags · When to involve a digital forensic expert

The Android deleted-data checklist

- ☐ Device preserved promptly; use minimised against overwriting and TRIM
- ☐ Achievable depth and encryption state assessed for this device
- ☐ File-system depth or deeper obtained where the device permits
- ☐ SQLite free space, WAL and journals recovered
- ☐ App trash and recently-deleted areas examined
- ☐ Unallocated space carved where physical depth allows; encryption/TRIM effect stated
- ☐ Backups, Google account and SD card used for pre-deletion states
- ☐ Recovered records reassembled; completeness stated
- ☐ Deletion dated where possible; records attributed with care
- ☐ Every "nothing deleted" claim read against its method's depth

Red flags

- A logical report's "nothing deleted" accepted: Example 3.
- Classic carving expected to work on an encrypted, TRIMmed device: Example 2.
- The device used after the deletion, overwriting remnants.
- Database remnants overlooked in favour of unallocated space: Example 1.
- A fragment presented as a whole record.
- The estate and SD card ignored when the device's limits block recovery.
- A failed carve left unexplained.

When to involve a digital forensic expert

Immediately, and before the phone is used further, because deleted Android data survives only until it is overwritten or TRIMmed, and only an expert can assess what the device's depth and encryption make reachable: the expert preserves the device and assesses the achievable depth and its encryption and storage (§4, §5), recovers deleted records from the SQLite free space, journals and app trash at file-system depth, where most cases are decided (Example 1, §3), and carves unallocated space where physical depth permits, explaining honestly the effect of file-based encryption and TRIM when it yields little (Example 2). Every claim that no deleted data exists is read against the method that produced it, and a deeper extraction sought where the device permits (Example 3, §4); where the device's limits block recovery, the pre-deletion backups, the Google account, the SD card and counterparts are used and the deletion dated (§8); and every recovered record is reassembled, its completeness stated, attributed with care and reported at honest confidence under CPR Part 35 or CrimPR Part 19. This guide continues the Android block. Through **cflab.uk** and **e-discovery.uk**, Android deleted-data work

recovers what the device still readably holds, explains what its modern protections have put beyond reach, and turns to the estate for what the device has lost.

§ 13 · COMMON QUESTIONS

Frequently asked questions

Can deleted data be recovered from an Android phone?

Often yes, but it depends on depth, encryption and timing (§2). Deleted records frequently survive in the free space and journals of the SQLite databases that hold Android's messages, calls and app data, and a file-system extraction reaches them, so deleted conversations are commonly restored. But a logical extraction sees no deleted data at all, and on modern devices file-based encryption and TRIM mean that carving unallocated space for deleted files works far less often than it once did. So recovery is a question of what depth the device permits, what its storage still readably holds, and how promptly it was preserved.

Why does the depth of extraction matter so much?

Because each depth reaches different deleted data (§4, Example 3, guide 162). A logical extraction takes only what the device hands over and never includes deleted records, so it recovers nothing deleted and its silence proves nothing. A file-system extraction reaches the databases and their free space and journals, where most deleted records survive. Only a physical extraction reaches unallocated space outside the databases. So what can be recovered is decided by the depth the device permits, and any claim that "no deleted data exists" is read against the method that made it.

What are file-based encryption and TRIM, and why do they matter?

The two modern limits on classic recovery (§5, Example 2). File-based encryption encrypts each file with its own key, so when a file is deleted and its key discarded, its remnants in unallocated space are unreadable ciphertext even to a physical extraction. TRIM tells flash storage which deleted blocks it may erase, and the storage often erases them soon after, so deleted data may be physically gone quickly. Together they mean that carving unallocated space on a modern Android device often yields little, so a failed carve is explained honestly and the estate is pursued instead.

So is deleted data on a modern Android phone unrecoverable?

No, because database remnants largely survive the modern limits (§5, Example 1). Deleted records inside a live database's free space and journals sit within a file that is still allocated and decrypted while the device is unlocked, so they survive encryption and TRIM in a way that unallocated data does not, and a file-system extraction reaches them. This is where most deleted-data cases are decided. What the modern limits mostly defeat is the carving of deleted files from unallocated space, and for those the pre-deletion backups, the Google account, an unencrypted SD card and counterparts are the routes.

Why does preserving the phone quickly matter?

Because two clocks are running (§2, §6). Continued use writes new data that reuses the free space where deleted database records linger, gradually overwriting them; and TRIM may erase deleted blocks in unallocated space within a short time regardless of use. So deleted data is recoverable "for now", and preserving the phone promptly, with its use minimised, maximises what survives to be recovered. Prompt preservation is often the difference between a full recovery of a deleted conversation and a lost one.

What if the phone genuinely no longer holds the deleted data?

Then the estate usually does (§8, Example 2). Local, manufacturer and Google backups hold earlier states of the databases with records from before the deletion, reachable regardless of the phone's lock; an unencrypted SD card holds its own carveable remnants free of the phone's encryption; the counterpart's device holds the other side of a deleted conversation; the carrier holds SMS and call records; and app providers hold their own. Comparing a pre-deletion backup with the live database can also date the deletion. So a device whose depth, encryption or TRIM has put the data beyond reach is rarely the end of the enquiry.

§ 14 · REFERENCE

Glossary

SQLite free space
Database pages marked free after deletion, holding old content.

Write-ahead log (WAL)
SQLite's log of recent changes.

Unallocated space
Storage outside files, holding deleted data.

Carving
Recovering files or records from unallocated or free space.

File-based encryption
Per-file encryption; deleted files' keys are discarded.

TRIM
Flash storage erasing blocks marked as deleted.

Logical / file-system / physical
The three depths of acquisition.

App trash
A holding area for deleted items before removal.

Pre-deletion backup
A backup made before a deletion, holding the data.

Overwriting
Reuse of free space that destroys remnants.

Sources and authoritative references

The Android deleted-data disciplines in this guide draw on materials published by our laboratory and e-discovery practice, referenced here as sources:

- cflab.uk, digital forensics services (SQLite and unallocated-space recovery, encryption- and TRIM-aware analysis, CPR Part 35 and CrimPR Part 19 reporting): cflab.uk/digital-forensics-services · guides library: cflab.uk/guides
- e-discovery.uk, EDRM Phase 04 · Collection and Phase 06 · Analysis (deleted-data recovery and reassembly as practised): e-discovery.uk/edrm/collection · e-discovery.uk/edrm/analysis
- e-discovery.uk, practice method and Knowledge Centre: e-discovery.uk · e-discovery.uk/knowledge
- SQLite documentation (file format, WAL) and Android file-based encryption documentation: sqlite.org, file format · sqlite.org, WAL · source.android.com, file-based encryption
- Forensic Science Regulator, Code of Practice v2 (recovery, reassembly and honest reporting) and ISO/IEC 27037: gov.uk, FSR Code · iso.org, 27037
- PD 57AD and CPR Part 31 · CPR Part 35: justice.gov.uk, PD 57AD · justice.gov.uk, Part 35

§ HOW A SPECIALIST LABORATORY CAN ASSIST

Working with Computer Forensics Lab

Android deleted-data work at the laboratory recovers what the device still readably holds, explains what its modern protections have put beyond reach, and turns to the estate for what the device has lost. The device is preserved from continued use, because overwriting reuses the free space where deleted database records linger and TRIM erases deleted blocks, and the acquisition depth it permits and its encryption and storage characteristics are assessed before any expectation is set (guides 162, 163). At file-system depth or deeper, deleted messages, calls and app records are recovered from the SQLite databases' free space, write-ahead logs and journals and from app trash, which is where most deleted-data cases are decided and where remnants largely survive the modern limits (Example 1, guides 155, 165, 167). Where physical depth is available, unallocated space is carved, and where file-based encryption and TRIM cause it to yield little, that effect is stated honestly rather than left unexplained (Example 2). Every claim that no deleted data exists is read against the method that produced it, and a deeper extraction is sought where the device permits (Example 3); where the device's limits block recovery, the pre-deletion local, manufacturer and Google backups, any unencrypted SD card and the counterparts are used and the deletion dated (guides 174, 164, 166). Every recovered record is reassembled, its completeness stated, attributed with care, corroborated and reported at honest confidence under CPR Part 35 or CrimPR Part 19. This guide continues the Android block. Conflicts checks, confidential scoping discussions and fixed-fee estimates are routine; and the guiding message is urgent and candid: deleted Android data is very often recoverable, but depth, encryption and TRIM decide the odds, and the phone must reach the laboratory before the clocks run out.

INSTRUCT THE LAB

Speak to a forensic examiner, not a salesperson.

Describe your matter in confidence and an examiner will respond the same working day. Conflicts checks and scoping calls with US counsel are routine.

NEW ENQUIRIES
+44 (0)20 7164 6971

EMAIL
info@cflab.uk

E-DISCOVERY
e-discovery.uk

Computer Forensics Lab Ltd · Euro House, 133 Ballards Lane, Finchley, London N3 1LJ, United Kingdom
cflab.uk · Customer service +44 (0)20 7164 6915 · ICO ZB395023 · NPCC · ISO 17025 aligned practice · UK Gov Digital Marketplace