IRONSIGHTS

Reading around ransomware

Recovering data and evidence from partially encrypted files.

A recovery and investigation case study for intermittently encrypted systems.

By **Ryan Balloot**

Microsoft Certified: Cybersecurity Architect Expert

IronSights, Sydney · Technical whitepaper · July 2026

How a business, and its evidence, were recovered from partially encrypted ransomware files, without paying and without breaking the encryption.

ABSTRACT

Organisations hit by ransomware are usually told that their only options are to restore from backup or to pay for a decryption key. When backups have failed and no key is available, the data is written off. This paper documents a different outcome, drawn from a real engagement, and the discipline required to reach it safely. The enabling weakness is partial encryption: to run quickly on large files, many ransomware families overwrite only the front of each file and leave the bulk of the payload intact. Nothing in this work breaks the cryptography. The recovery reads around the damage rather than attacking the mathematics, exploiting a weak implementation rather than a weak cipher. We first establish which regions of a file survive, by analysing the encryptor's behaviour and by validating that the file's expected structures are intact at their computed offsets, then recover the surviving data using the format's own structure: page-level carving reconstructs a database from its intact pages, while an encrypted virtual disk whose header was overwritten but whose filesystem was not is recovered by computing the layout and reading the intact filesystem directly. We then argue the paper's central point: a recovery that reports success is not a verified recovery. Carving-based tools return output that looks complete but silently is not, and only validation against a point-in-time baseline separates real data from artefacts. Finally, because the same reading of intact structures recovered the system logs, the effort became an investigation as well as a recovery, and we describe the evidentiary discipline that made its findings defensible: multi-source corroboration, and a strict separation of what was confirmed from what was only indicated. We set out the techniques, their results, and their honest limits.

Keywords: ransomware, intermittent encryption, partial encryption, data recovery, data validation, incident response, digital investigation, indicators of compromise.

1. Introduction

The prevailing response guidance for a ransomware incident offers a short list of options: restore from a clean backup, use a public decryptor if one exists, rebuild from source systems, or pay. When backups are absent or themselves encrypted, no decryptor exists, and source systems cannot reproduce the data, organisations are commonly told that recovery is impossible without the attacker's key. Payment then looks like the only path, despite carrying no assurance that a key will be provided or will work.

This paper documents a recovery that succeeded where that advice would have counselled payment, and it does two things beyond describing the technique. First, it argues that the hard part of this work is not the recovery but the validation: a recovery tool that reports success will readily hand back a corrupt and incomplete result that fails silently if trusted. Second, it shows that recovering the intact structures of an encrypted system also recovers its evidence, so that a recovery and an investigation can be the same effort, and it sets out the discipline that keeps the investigation's conclusions defensible.

This is a case study, not a claim of a new primitive. Partial encryption and the individual recovery techniques are documented in both the security research and the commercial data-recovery literature, and prior practitioner work has recovered event logs from ransomware-encrypted virtual disks and used them for investigation. Its value is in the completeness and honesty of an end-to-end record of a hard real case, and in two disciplines that are widely practised but rarely written up, not in novelty.

1.1 Prior work, and how this differs

The enabling weakness, intermittent or partial encryption, has been measured and modelled in the research literature and reported by vendors since 2022. Commercial data-recovery providers describe, in general terms, recovering usable data from partially encrypted large files and databases. Closest to the virtual-disk work here, a practitioner write-up has recovered Windows event logs from ransomware-encrypted virtual disks by signature-based fragment carving and used them for investigation.

This case differs in three concrete ways. First, the virtual disk is recovered not by fragment carving but by computing the container's layout and reading the intact filesystem structurally, which returns whole files rather than the partial and sometimes incomplete results of signature carving, and is possible here because only the header was overwritten. Second, it adds the recovery of a large database by page-level carving, with the validation that carving demands. Third, it holds the recovered evidence to a defensible standard: separating the confirmed from the indicated, and corroborating the central negative from independent sources. None of these is a new technique in isolation. Together they are a fuller and more honest account than the individual pieces have received.

2. Background: partial encryption, and reading around the damage

> **Nothing in this work defeats the cryptography. The recovery reads around the damage, not through it.**

Encrypting a large file in full is slow. To move quickly and finish before intervention, many ransomware families encrypt only part of each file. This is variously called intermittent, partial, or chunked encryption, and public analysis has documented its adoption across multiple families as a deliberate trade-off between speed and thoroughness. Common patterns include overwriting only the first portion of a file, or encrypting fixed-size blocks at intervals.

The consequence for recovery is that a speed-optimised attack leaves large regions of the original file untouched. The file will not open, because the parts that were overwritten are often structurally important, such as a header or a file's leading pages, but most of the bytes are unmodified.

This frames the recovery philosophy plainly, and it is worth stating because it is easily misread. **Nothing in this work defeats the cryptography.** The encrypted regions are left exactly as they are. The recoveries succeeded because the implementation was weak, partial encryption on large files, and because the response read around the damaged regions rather than attacking the encryption itself. This is not a break of ransomware encryption, and it should not be reported as one. It is the recovery of data that the attacker never encrypted.

The approach is most exploitable on large files with predictable internal structure: database files, virtual disks, archives, and large media. These formats are laid out according to known specifications, with identifiable headers, fixed-size blocks, and computable offsets, so the intact regions can be located and read precisely because their structure is predictable.

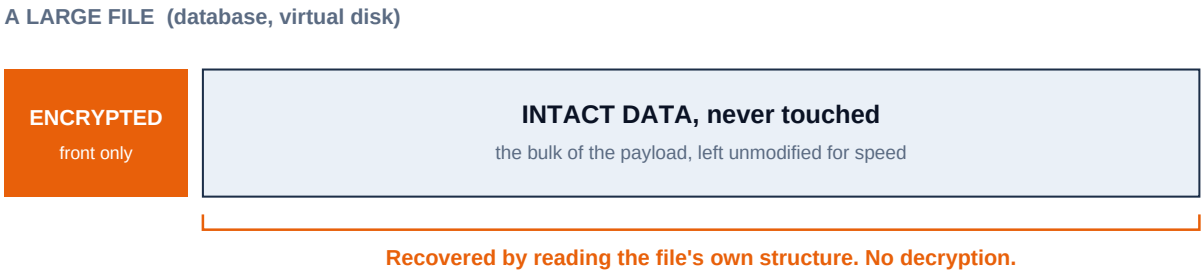

Figure 1. Ransomware encrypts only the front of a large file for speed, leaving most of the data intact. The recovery reads that intact remainder using the file's own structure. It does not break the encryption.

3. Recovery methodology

The work is described at the level of principle. It is not a tool, and the intent is to document the approach rather than to provide a turnkey capability. What links the cases is a single weakness, not a single technique: the technique used to exploit partial encryption differs by the type of artefact.

A note on tooling, because it bears on credibility. In the engagement described in Section 5, the encrypted Windows virtual disks, their NTFS filesystems, the registry, the Windows event logs, and the domain's identity database were all examined on macOS using open-source tooling, with no Windows agent installed and no decryption performed, by computing disk layout and reading intact structures directly. The method is not tied to the operating system that produced the data.

3.1 Establish which regions survived

Before any recovery, the surviving regions of a file must be identified. Several methods can do this, and they are complementary.

- **Encryptor behaviour analysis.** Static analysis of the ransomware itself establishes its encryption pattern: for the family in this engagement, that it overwrites only the front, or a limited leading region, of large files. This tells you where the damage lands, and therefore where intact data should remain, without scanning the victim files at all. The same analysis also confirmed the encryption was sound: a hybrid construction in which AES-256-CBC encrypts the file content and each file's AES key is itself RSA-wrapped, so no symmetric key was recoverable, which is precisely why the recovery reads around the damage rather than attempting to defeat it.
- **Structural signature validation.** The presence of intact data is confirmed directly, by finding the file format's known structures at their expected offsets. For a container file, that means locating a valid partition-table signature, filesystem identifier, and file-table record at their computed locations; for a database, valid page structures at the fixed page boundaries. A structure that parses cleanly at the offset where it belongs is intact.
- **Entropy profiling.** A general, tool-agnostic assessment that measures randomness across a file: encrypted regions read as near-random (high Shannon entropy), intact structured data much lower. It is a legitimate way to confirm partial encryption and to locate encrypted versus intact regions, and is noted here for completeness, though in this engagement survival was established by the two methods above rather than by an entropy map.

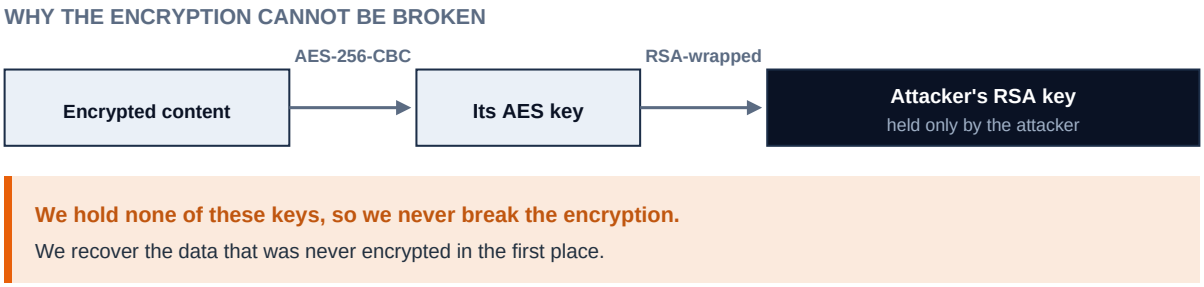

Figure 2. The encryption is sound: file content is encrypted with AES-256-CBC, and each file's AES key is itself wrapped with the attacker's RSA key. No key is recoverable, which is why the recovery reads around the damage rather than attacking the cipher.

3.2 Recover using the format's own structure

How the surviving data is recovered depends on the artefact. Two distinct techniques appeared in the engagement.

- **Page-level carving**, for a structured data file such as a database. The recovery scans the file for the intact fixed-size pages that survived, reads the data directly from them, and reconstructs the database from those pages while skipping the encrypted regions. This is genuinely carving: the surviving pages are found and lifted out of a partially damaged file.
- **Offset-computed structural read**, for a container file such as a virtual disk. Here there is no carving. Because the filesystem inside the container was never touched and only the container's header was overwritten, the fixed layout is computed to seek past the damaged header, and the intact filesystem is then read directly through a reader that offsets every access. The files inside come out as ordinary files. Nothing is reconstructed from fragments; the intact filesystem is simply read where it still sits.

In some incidents a host is not encrypted at all, in which case its data is read directly, with no bypass of any kind.

3.3 Rebuild

Recovered data is assembled into a usable form, for a database a fresh, structurally sound database on a clean server. The output is a working artefact built from the surviving data, not an in-place decryption of the original. What it is not yet is verified, which is the subject of the next section.

4. Validation: a recovery is not a verified recovery

> **The dangerous failure mode is not a recovery that errors. It is one that succeeds and lies.**

This is the part the tooling will not do for you, and it is where the real risk lies. A recovery tool reports success when it has produced output, not when that output is correct. Carving-based recovery in particular will hand back a database that looks full and is quietly wrong.

In this engagement the carved output looked complete and was not. Because carving reads fixed-size pages wherever it finds them, it mis-attributes fragments that happen to resemble valid pages. The recovered database contained fragments of the database engine's own internal structures where user data should have been, values sitting in the wrong columns, braced identifiers appearing as data, spurious columns injected into table headers, and null values exported as the literal text of the word rather than as true nulls. Each of these passes a casual eye. Handed to the client unchecked, the result would have been a database quietly missing core records while appearing intact.

Two checks caught it, and both are essential.

- **Row-count reconciliation against a point-in-time baseline.** Every table's recovered row count is compared against a known-good reference from before the incident. A table that should hold a known number of records and holds fewer has lost data, however complete it looks.
- **Structural corruption detection.** Each table is inspected for signs that carving has gone wrong: are the column names real, are the values aligned to the columns that should contain them, does a field that should hold a code instead hold a date or a fragment of something else. Structure, not just row count, reveals mis-carved data.

The lesson generalises beyond this case. Any recovery built on carving or fragment reassembly should be treated as unverified until reconciled against an independent baseline. Reporting recovered data without that step is not recovery; it is the appearance of recovery, which is worse, because the client trusts it.

5. The engagement

An organisation was hit by a ransomware attack from the Makop/Phobos family, which appended the `.ndm448` extension to encrypted files. By the time we were engaged, every conventional recovery path had already failed: the offline tape backups were corrupt, the on-site backups had themselves been encrypted, and the volume shadow copies had been deleted. A core business database, comprising millions of records across more than a thousand tables, was inaccessible, and a ransom had been demanded with no assurance that payment would restore the data.

5.1 The three recovery mechanisms

The recoveries all exploited the same weakness, partial encryption, but each artefact required a different technique. Three are worth separating, because conflating them would misrepresent the work.

- **The business database (page-level carving).** The core database file was partially encrypted. Using page-level carving, we scanned the file and reconstructed the database from its intact, unencrypted pages, skipping the encrypted regions. After the validation described in Section 4, this returned the majority of records, on the order of three-quarters of the pre-incident baseline, without the attacker's key and without payment, with the remaining shortfall concentrated in an identified set of tables flagged for further recovery, enabling the organisation to rebuild on a clean system.
- **The domain controller's virtual disk (offset-computed structural read).** The domain controller's disk, a large virtual disk file, had also been encrypted, but the ransomware had overwritten only the header at the very front. The filesystem deeper in the disk was never touched. Rather than carve, we computed the fixed layout by hand and confirmed each structure at its expected offset: a valid partition-table header signature at four mebibytes plus 512 bytes, the filesystem identifier at the partition offset (file offset 349,175,808), and a valid first file-table record. With the layout confirmed, we read the intact filesystem directly through a reader that offsets every access. The event logs, registry, and other system files came out as ordinary files, with no decryption. Those recovered logs were what allowed us to establish the indicators of compromise and reconstruct how the attack had unfolded.
- **The initial foothold (read directly).** One host, the machine the attacker had used as its initial foothold, had never been encrypted at all. Its logs were read directly from an intact disk, with no bypass of any kind. Together with the domain controller's recovered logs, this completed the picture of the intrusion.

Applied to the encrypted virtual disk, the method did not only return data; it restored the very evidence needed to understand the intrusion. Recovery and investigation, usually treated as separate efforts, became one.

5.2 Establishing what happened, defensibly

Recovering the logs is not the same as knowing what they mean, and an investigation that overstates its certainty is worse than none. Three disciplines kept the findings defensible.

Separate what is confirmed from what is only indicated. Every conclusion was labelled as either proven from evidence or as a capability that was present but not positively confirmed, with the reason stated. Where process-creation auditing had been disabled, for example, the execution of a tool could not be positively logged, so its use could not be confirmed even where the tool was present. The correct posture in that case is to assume compromise and scope to the worst plausible case. Logging gaps do not shrink the incident; they enlarge it, because absence of evidence is not evidence of absence when the evidence was never being recorded.

Corroborate a negative from independent sources. The ransom note claimed bulk theft of data. Rather than accept or dismiss the claim, we tested it, and a defensible "no bulk exfiltration occurred" required more than one measurement. Four independent sources agreed: the firewall's flow logs, the firewall hardware's own interface byte-counters, the off-box uplink meter at the internet provider, and the absence of any staging archives on the hosts that theft would have required. One source is an opinion; four independent sources that agree is a finding. This is what let the organisation answer the note's extortion claim with evidence rather than fear.

Trace the blast radius honestly. The intrusion showed how a single weakness becomes a total compromise. A reused password pattern meant that the credential captured for a privileged domain account was also the master password to the IT team's password vault, so the entire keychain for the infrastructure fell with one credential. Control of the domain controller in turn exposed the domain's identity database. The lesson is not subtle, but it is worth stating plainly: password reuse is a blast-radius multiplier, and a single reused credential can be the difference between an incident and a catastrophe.

6. Results and honest limits

The approach can return partial to substantial recovery, in favourable cases a substantial proportion of records, roughly three-quarters in this engagement, without paying and without the attacker's key, and it can recover the evidence needed to understand the intrusion. It is strongest where organisations are most exposed: large, business-critical structured files and system disks.

Its limits should be stated plainly, because overstating outcomes would undermine the credibility the method deserves.

- It depends on the ransomware using partial or intermittent encryption. Some families encrypt files fully, and where that is so the approach does not apply.
- Small files, more likely to be encrypted end to end, recover less reliably than large ones.
- It is a recovery effort, not a guarantee. Results are reconciled and validated, and require the data owner's or software vendor's sign-off before production use.
- The investigation is bounded by what was logged. Where auditing was disabled, some questions can only be answered by assuming the worst.

7. Prevention: foreseeable, and foreseen

The controls that would have prevented or contained this incident are unremarkable, which is the point. Internet-exposed remote desktop access gave the initial foothold; reused local administrator credentials let it spread; the absence of multi-factor authentication removed the last barrier to the privileged accounts; and the absence of endpoint detection meant the activity ran unobserved. Each is a well-understood control.

What makes the case instructive is that the path was not only foreseeable but foreseen: a prior security assessment had predicted this attack path (reused local administrator credentials, absent multi-factor authentication, and no endpoint detection), and had it been remediated, the incident would very likely have been prevented or contained. The recovery succeeded, but it should never have been needed. The strongest argument in this paper for basic controls is that the alternative is to depend on a weak implementation of ransomware encryption and on the availability of a validated recovery, neither of which is a plan.

8. Reporting obligations (an example jurisdiction)

Recovery decisions sit alongside legal and evidential obligations that vary by jurisdiction. Using Australia as a worked example, and as general guidance rather than legal advice: unauthorised access to personal information may require assessment and potential notification of the regulator and affected individuals under the Notifiable Data Breaches scheme, and a separate obligation to report a ransomware payment applies to larger entities under the Cyber Security Act 2024. The general principle travels even where the specific statutes do not. Reporting duties commonly attach to unauthorised access rather than to proven exfiltration, they can apply independently of whether data was recovered, and preservation of evidence in the first hours is what makes both the recovery and the reporting possible. Organisations should seek advice on the obligations of their own jurisdiction.

9. Conclusion

When backups have failed and no key is available, ransomware-encrypted data is commonly declared unrecoverable and payment treated as the only option. For the growing class of files that ransomware encrypts only partially, that conclusion is often wrong. By establishing which regions of a file survive and reading them through the file format's own structure, without ever breaking the encryption, a large structured file can be returned without the attacker's key, and the same reading of intact structures can recover the evidence that explains the intrusion. The techniques vary with the artefact, but the enabling weakness, and the honesty required, are constant. The recovery must be validated against an independent baseline or it is only the appearance of recovery. The investigation must separate the confirmed from the indicated, and corroborate its negatives, or its conclusions cannot be relied upon. Held to that standard, ransomware recovery is a technical problem to be worked and evidenced, not a payment to be made.

References

1 SentinelLabs, "Crimeware Trends: Ransomware Developers Turn to Intermittent Encryption to Evade Detection" (2022).

2 "Intermittent File Encryption in Ransomware: Measurement, Modeling, and Detection," arXiv:2510.15133 (2025).

3 BleepingComputer, "Ransomware gangs switching to new intermittent encryption tactic."

4 The DFIR Journal, "File Carving: Encrypted Virtual Hard Disks."

5 FBI, CISA, and MS-ISAC, "#StopRansomware: Phobos Ransomware," Joint Cybersecurity Advisory AA24-060A (2024).

6 Shannon, C. E. "A Mathematical Theory of Communication" (1948).

7 Australia, Privacy Act 1988 and the Notifiable Data Breaches scheme; Cyber Security Act 2024.

How to cite. Balloot, R. (2026). *Reading around ransomware: recovering data and evidence from partially encrypted files.* IronSights, Sydney.

About the author. Ryan Balloot is the managing director of IronSights, a Sydney cyber security firm he founded out of a passion for securing small business, and a Microsoft Certified: Cybersecurity Architect Expert. Contact hello@ironsights.com.au or 1300 004 766.