International Journal of Computational Research in Science and Technology
E-ISSN: 3108-0871, Volume 2 Issue 5
Sep-Oct 2026, PP 01-8

Digital Forensics Using Memory and File-System Analysis: A Critical Review of Acquisition, Artefact Analysis and Correlation Techniques, with a Conceptual Integrated Architecture

Waghole Krushna Subhash, Prof. Bangar Abhishek Popat, Prof. S. B. Bhosale, Prof. Karad V. A.
Dept. of Computer Engineering,
Jaihind collage of Engineering, Pune,India

Abstract. Evidence on a computer lives in two broad places. Volatile memory (RAM) captures what the system is doing right now, while persistent storage, through its file system, keeps a long-running record of what was created, edited, moved and erased. RAM can hold running processes, open network connections, loaded modules, injected code, credentials and encryption keys, much of which never touches the disk. File systems, in turn, hold files, metadata, journals, timestamps and traces of deleted content. This paper is a structured narrative review of 26 sources (standards, books and peer-reviewed studies) grouped under six themes: forensic principles and process models; memory acquisition; memory analysis and artefact recovery; file-system analysis, carving and timelines; solid-state storage and anti-forensics; and scale and tool validation. The literature shows that memory analysis can uncover hidden processes, injected code and key material that a disk examination would miss, whereas file-system analysis rebuilds history and recovers deleted content that memory cannot offer. It also shows that acquisition itself introduces error, because capturing memory changes the machine being captured. Yet the two sources are mostly studied apart, acquisition tools can be evaded or may yield inconsistent images, SSD wear-levelling and TRIM weaken the assumptions behind classical recovery, and growing data volumes are overwhelming manual workflows. The review offers a thematic synthesis, a comparison of twelve representative studies, a conceptual seven-layer architecture that correlates memory and file-system evidence within an integrity-preserving workflow, and a prioritised research agenda.

Keywords: Digital Forensics; Memory Forensics; File-System Analysis; Volatile Memory Acquisition; Data Carving; Timeline Analysis; Anti-Forensics; Evidence Integrity

I. Introduction

Digital forensics is the practice of identifying, preserving, analysing and presenting digital evidence in a way that is dependable and acceptable in court. Early research agendas already called for shared process models, validated tools and more attention to volatile data [1], and a later look back at the field argued that rising data volumes, encryption and a widening range of devices would keep testing those foundations [2].

Two sources of evidence dominate host-based investigations. Persistent storage is examined through the file system, which organises files, directories, allocation details and timestamps; with its structures an examiner can list files, recover deleted entries and build a timeline [6]. Volatile memory is examined by rebuilding operating-system data structures from a physical

memory image, exposing the processes, sockets, drivers and loaded code that existed when the capture was taken [7]. Collection guidance has long advised gathering volatile data first, since it disappears at power-off [3], and incident-response guidance treats memory and file-system collection as parts of one workflow [4].

The traditional "dead-box" approach of analysing a powered-off disk is no longer enough. Malware can live entirely in memory, full-disk encryption can leave a disk unreadable unless the keys held in RAM are obtained, and cloud or containerised workloads may leave very little on local storage. At the same time, a memory image alone cannot explain what happened over weeks or months. The two sources therefore complement each other, even though they are usually taught, tooled and researched separately.

This review has six aims: (i) to explain the principles and process requirements behind forensic acquisition; (ii) to survey memory acquisition and analysis techniques; (iii) to survey file-system analysis, carving and timeline methods; (iv) to look at solid-state storage, anti-forensics and scale as open challenges; (v) to propose a literature-based conceptual architecture that correlates both evidence sources; and (vi) to point out research gaps.

II. Review Methodology

The study is a structured narrative review. It draws on standards and guidance documents (RFC 3227, NIST SP 800-86, ISO/IEC 27037), reference books, and peer-reviewed papers from venues such as Digital Investigation, the DFRWS conferences, and ACM and IEEE outlets. Search themes combined memory forensics, volatile memory acquisition, Volatility, process and module analysis, file-system forensics, NTFS, ext4, APFS, data carving, timeline analysis, solid-state drives and anti-forensics. Foundational works were kept where they introduced a method or abstraction still in use, and newer work was preferred where it was available.

Logged searches of databases such as Scopus, Web of Science and IEEE Xplore were not carried out for this version, so the review is narrative rather than systematic and no study counts are given. Any quantitative statement is limited to what the cited sources themselves report. Anyone extending this work should repeat the search strings in those databases, check every reference against the publisher's record, and add recent (2022–2026) studies to the comparative tables.

III. Forensic Principles and Process Models

1. Order of volatility and integrity

RFC 3227 lays out the order of volatility: registers and cache first, then memory, network state and running processes, and finally disk and archival media. It tells the investigator to collect the most volatile evidence first and to document every step [3]. NIST SP 800-86 describes a four-phase process of collection, examination, analysis and reporting, and treats memory and file-system data as sources within a single incident-response workflow [4]. ISO/IEC 27037 adds requirements for identifying, collecting, acquiring and preserving potential digital evidence, including the competence of the person handling it and a documented chain of custody [5]. Cryptographic hashing of acquired images remains the standard integrity control.

2. Layered abstraction for tools

Carrier suggested that forensic tools be understood through layers of abstraction. A tool takes input data, applies rules and produces output with a known error rate, and each layer (bytes, volume, file system, application) can add errors of its own [26]. The idea applies equally to memory and disk analysis: a file listing or a process list is an interpretation of lower-level data, not raw truth, so an examiner has to be able to check that interpretation against the layer beneath it.

IV. Memory Forensics

1. Acquisition

Acquiring memory is invasive by nature. Software-based acquisition runs code on the live machine, so the tool itself occupies memory and alters system state, and the target operating system may interfere with the capture. Schatz's BodySnatcher improved reliability by injecting an independent acquisition environment instead of depending on an operating system that might be compromised [10]. Stüttgen and Cohen showed that ordinary acquisition can be fooled by anti-forensic manipulation and proposed an approach that holds up against such tampering [12]. Hardware and firmware methods do not rely on the target operating system, but they need physical access and prior preparation.

Physical remanence offers another route to memory, and a security risk at the same time. Halderman et al. demonstrated that DRAM contents can persist for seconds to minutes after power is lost, and for longer when the chips are cooled, which made it possible to recover encryption keys after a reboot or power interruption [11]. For investigators this means memory evidence can survive brief interruptions; for defenders it shows that keys sitting in RAM are exposed.

Mobile platforms make things harder. Sylve et al. described how to acquire volatile memory from Android devices with a loadable kernel module and analyse it using the Volatility framework, while noting how much effort is needed to adapt the tools to device-specific kernels [14].

2. Analysis and artefact recovery

Memory analysis rebuilds high-level objects from a flat image by relying on knowledge of operating-system structures. FATKit was an early framework that extracted and interpreted data from volatile memory using structure definitions [8]. Dolan-Gavitt showed that the Virtual Address Descriptor tree in Windows gives a process-level view of memory regions, mapped files and protections, which helps detect injected code [9]. The Volatility framework, documented in The Art of Memory Forensics, brought together plugins for process listing, hidden-process detection by cross-checking several kernel lists, network connection recovery, loaded drivers and modules, in-memory registry hives, command history and extraction of cached files [7].

Several kinds of artefacts can be recovered from memory yet are often missing from disk: processes that have been unlinked from the active list, injected or reflectively loaded code that never reached the disk, decrypted content and keys, recent network connections, clipboard and command-line contents, and credentials sitting in process memory. Block and Dewald studied page-table entries to find injected code that was hidden, whether on purpose or by accident, in Windows, which illustrates the shift from list-based detection to detection based on memory layout [15].

3. Limitations of memory forensics

Case and Richard's assessment of the "path forward" listed several recurring problems: operating-system internals change quickly and break analysis tools, profile or symbol information is needed for each kernel version, large-memory systems are difficult to handle, and memory-resident or fileless attacks are growing, which makes memory analysis more important rather than less [13]. A memory image is also only a single moment. It can be smeared because the system keeps running during capture, and data that has been paged out is not in it.

V. File-System Forensics

1. Structures and recovery

Carrier's treatment of file-system forensics is the standard reference. The examiner works through layers of file-system structure (metadata, content, file-name and application categories) to list files, recover deleted entries whose metadata survives,

and interpret timestamps [6]. Whether a deleted file can be recovered depends on whether its metadata and content areas have since been reused.

Newer file systems have changed what evidence exists. Fairbanks analysed ext4 for forensic purposes, covering extents, journalling and delayed allocation and what they mean for timestamps and the recoverability of deleted files [16]. Hansen and Toolan decoded the Apple File System (APFS) and documented its container and volume structure, snapshots and copy-on-write behaviour, all of which matter for both evidence recovery and timestamp interpretation [17]. NTFS offers rich metadata through its master file table, change journal and log structures, but interpretation needs care because several timestamp sets exist and routine system activity updates them.

2. Carving

Once metadata is gone, content-based recovery (carving) scans raw data for file signatures. Plain header/footer carving breaks down on fragmented files. Garfinkel proposed carving both contiguous and fragmented files with fast object validation, which tests candidate fragment combinations against the structure of the file type to cut down false positives [19].

3. Hashing and fingerprinting

Hashing whole files can identify known files, but it fails once a file has been modified or partly overwritten. Context-triggered piecewise hashing was introduced to spot almost-identical files [21], and block-hash approaches make it possible to identify small blocks and sub-file content even in fragmented or partly recovered data [20]. Roussev's survey of hashing and data fingerprinting explains how approximate matching supports triage and similarity analysis [22].

4. Timeline analysis

A timeline merges file-system timestamps, log entries and application artefacts into one chronology. Its strength is reconstructing the order of events; its weakness is that timestamps can be altered, differ in granularity between file systems, and sometimes reflect what the system did rather than what a user did. Timeline claims therefore need support from independent sources, and memory evidence is a natural candidate for that role.

VI. Solid-State Storage, Anti-Forensics and Scale

1. Solid-state drives

Bell and Boddington argued that solid-state drives undermine classical recovery assumptions. Wear-levelling, garbage collection and TRIM mean the device may erase deleted data on its own schedule, and repeating the same acquisition can give different results [18]. Carving and deleted-file recovery results from hard disks therefore cannot be assumed to carry over to flash storage.

2. Anti-forensics

Harris's analysis of anti-forensics sorted attacks on evidence into destruction, hiding, elimination of sources and counterfeiting, and called for a shared understanding of the problem [25]. Memory-resident techniques (such as unlinking processes) and file-system techniques (such as timestamp manipulation and secure deletion) both aim to defeat the analysis layers described above. The answer is to cross-check independent sources, as in multi-list process detection [7] and tamper-resilient acquisition [12].

3. Scale

Quick and Choo surveyed how growing data volumes affect digital forensics and identified rising storage capacity, device diversity and case backlogs as pressures that favour triage, data reduction and automation [23]. Memory images of several gigabytes per host, collected across many hosts, add to the same pressure.

International Journal of Statistics and Mathematical Research Trends
E-ISSN: 3108-0871, Volume 2 Issue 5
Sep-Oct 2026, PP 01-8

VII. Comparative Analysis

Table I compares twelve representative studies and references. The entries describe contributions qualitatively; the studies target different systems and use different evaluation methods, so they cannot be compared directly.

Table 1: Comparison of Representative Studies in Memory and File-System Forensics

Author, year	Objective	Technique	Main contribution	Limitation
Carrier, 2003 [26]	Describe how forensic tools can be modelled	Layered abstraction	Errors and checks at every layer	Conceptual only; no benchmark of tools
Carrier, 2005 [6]	Standard reference on file systems	Analysis of on-disk structures	Established method for recovery and metadata reading	Written before APFS and newer ext4 features
Petroni et al., 2006 [8]	Examine volatile memory	FATKit framework	Extraction driven by structure definitions	Limited to early OS versions
Dolan-Gavitt, 2007 [9]	Per-process view of memory	VAD tree inspection	Spots mapped regions and injected code	Windows only
Schatz, 2007 [10]	Dependable memory capture	Separate acquisition environment	Less reliance on the target OS	Prototype; tied to one platform
Garfinkel, 2007 [19]	Restore fragmented files	Carving plus object validation	Fewer false matches	Depends on file type
Halderman et al., 2009 [11]	Test how long DRAM keeps data	Cold-boot tests	Encryption keys can be pulled from RAM	Attack-focused; hardware dependent
Fairbanks, 2012 [16]	Forensic study of ext4	Structural analysis	Journal, extents and timestamps covered	Linux only
Sylve et al., 2012 [14]	Memory analysis on Android	Kernel-module capture with Volatility	Shows mobile acquisition is workable	Effort needed per kernel
Stüttgen & Cohen, 2013 [12]	Withstand anti-forensics	Tamper-resilient acquisition	Capture that detects tampering	Narrow threat model
Hansen & Toolan, 2017 [17]	Decode the APFS format	Format reverse-engineered	Snapshots and copy-on-write described	Format keeps changing
Block & Dewald, 2019 [15]	Find hidden code	Page-table entry inspection	Goes beyond list-based detection	Windows only; details to verify

1. Thematic synthesis

Four patterns stand out. (i) Each source answers different questions. Memory shows live state, hidden activity and keys [7], [11], while file systems show history, deleted content and timestamps [6], [16]. (ii) Acquisition shapes the evidence. Capture tools alter system state, can be evaded, and on SSDs may see different data over time [10], [12], [18]. (iii) Tools depend on

versions. Memory analysis and file-system parsing both break as operating systems and formats evolve [13], [17]. (iv) Correlation is under-formalised. Studies of memory and of file systems rarely define how a finding from one source should corroborate the other, even though the integrity of a timeline depends on exactly that.

VIII. Proposed Conceptual Architecture

The architecture below is conceptual. It has not been implemented or tested experimentally, and it is derived from the requirements and weaknesses identified in the sections above.

Layer 2 follows the order of volatility: memory is captured before disk, and the acquisition method is recorded because it influences the evidence [3], [10]. Layer 3 logs hashes and tool versions, which lets results be reproduced and defended [5]. Layer 4 runs memory and file-system parsers independently and retains their raw output. Layer 5 is the distinguishing layer. It expresses artefacts from both sources as time-stamped events and looks for agreement and conflict. For instance, a process found in memory with no matching file on disk points to memory-resident code, and a file whose timestamp has been altered but whose journal entry disagrees points to manipulation [25]. Layer 6 reports each finding with the layer it came from and its known limitations, in line with the abstraction principle [26]. Layer 7 closes the loop: parsers and tools are checked against reference images before anyone relies on their output.

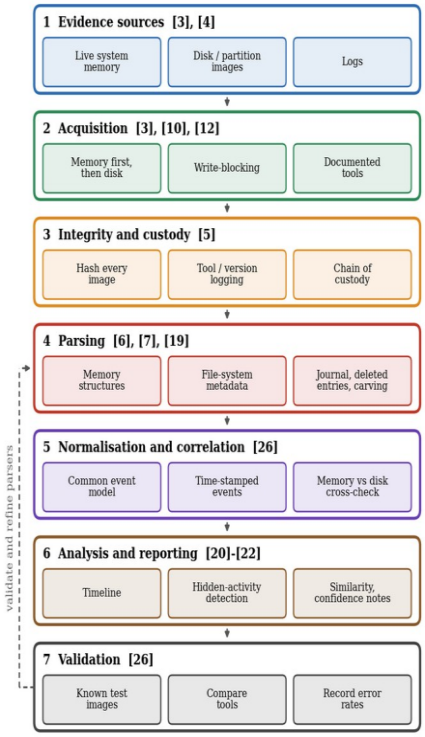


Fig. 1. Proposed conceptual seven-layer architecture for integrated memory and file-system forensic analysis (conceptual, not experimentally validated).

Research Gaps and Future Directions

Table II maps each gap to the evidence that supports it and to the corresponding future direction.

International Journal of Statistics and Mathematical Research Trends
E-ISSN: 3108-0871, Volume 2 Issue 5
Sep-Oct 2026, PP 01-8

Table 2: Research Gaps and Corresponding Future Directions

Gap	Supporting evidence	Future direction
Memory and disk treated apart	Two separate bodies of work [6], [7]	Formal correlation models with a common event schema
Acquisition changes evidence or can be evaded	[10], [12]	Acquisition that resists tampering, leaves a small footprint and can be verified
Tools tied to specific versions	[13], [17]	Automatic structure discovery and symbol handling
Fileless and memory-resident attacks	[13], [15]	Memory-layout detection instead of list-based checks
Uncertainty from SSDs, TRIM and wear-levelling	[18]	Recovery experiments on current flash devices
Reliability of timestamps	[25]	Validating timestamps across sources
Data volume and case backlog	[23]	Triage, similarity matching and scalable analysis [22]
Encrypted storage and keys held in RAM	[11]	Key-recovery workflows that stay within legal and ethical limits
Validation of tools and their error rates	[26]	Shared reference images and benchmark suites

Near-term work is to build a prototype of the correlation layer on reference images and report its error rates. Medium-term work is to extend correlation to cloud, container and mobile evidence. Longer-term directions include automated, verifiable acquisition that stays robust against adversaries, and agreed benchmarks for memory and file-system tools.

Limitations of this Review

This is a narrative review without logged database searches, so relevant studies may have been missed, particularly recent (2022–2026) work on cloud, container and mobile forensics. Performance figures are not reported because the cited sources use evaluations that cannot be compared. Before submission, every reference should be verified against the publisher's record and DOIs should be added.

IX. Conclusion

Memory analysis and file-system analysis answer different questions. Memory reveals what a system was doing and holding at a given moment, including activity and keys that never reach the disk, while file systems reveal what existed and how it changed over time, deleted content and metadata included. The literature describes mature techniques for each, but it also shows that acquisition affects evidence, that tools are tied to specific operating-system and format versions, that SSDs weaken the assumptions behind traditional recovery, that anti-forensic techniques go after both sources, and that no formal way of correlating memory and disk evidence has yet matured. A dependable workflow therefore needs acquisition in order of volatility, hash-based integrity and custody records, layered and validated parsing, explicit correlation of the two sources, and honest reporting of known error. The proposed architecture reflects these needs and should be validated on reference images and in real investigations.

References

1. G. Palmer, “A road map for digital forensic research,” DFRWS Tech. Rep. DTR-T001-01, Utica, NY, USA, 2001.
2. S. L. Garfinkel, “Digital forensics research: The next 10 years,” Digital Investigation, vol. 7, pp. S64–S73, 2010.
3. D. Brezinski and T. Killalea, “Guidelines for evidence collection and archiving,” RFC 3227, IETF, 2002.
4. K. Kent, S. Chevalier, T. Grance, and H. Dang, “Guide to integrating forensic techniques into incident response,” NIST SP 800-86, 2006.
5. ISO/IEC 27037:2012, “Guidelines for identification, collection, acquisition and preservation of digital evidence,” 2012.
6. B. Carrier, File System Forensic Analysis. Boston, MA, USA: Addison-Wesley, 2005.
7. M. H. Ligh, A. Case, J. Levy, and A. Walters, The Art of Memory Forensics: Detecting Malware and Threats in Windows, Linux, and Mac Memory. Indianapolis, IN, USA: Wiley, 2014.
8. N. L. Petroni Jr., A. Walters, T. Fraser, and W. A. Arbaugh, “FATKit: A framework for the extraction and analysis of digital forensic data from volatile system memory,” Digital Investigation, vol. 3, no. 4, pp. 197–210, 2006.
9. B. Dolan-Gavitt, “The VAD tree: A process-eye view of physical memory,” Digital Investigation, vol. 4, pp. S62–S64, 2007.
10. M. Schatz, “BodySnatcher: Towards reliable volatile memory acquisition by software,” Digital Investigation, vol. 4, pp. S126–S134, 2007.
11. J. A. Halderman et al., “Lest we remember: Cold-boot attacks on encryption keys,” Commun. ACM, vol. 52, no. 5, pp. 91–98, 2009.
12. J. Stüttgen and M. Cohen, “Anti-forensic resilient memory acquisition,” Digital Investigation, vol. 10, pp. S105–S115, 2013.
13. A. Case and G. G. Richard III, “Memory forensics: The path forward,” Digital Investigation, vol. 20, pp. 23–33, 2017.
14. J. Sylve, A. Case, L. Marziale, and G. G. Richard, “Acquisition and analysis of volatile memory from Android devices,” Digital Investigation, vol. 8, no. 3–4, pp. 175–184, 2012.
15. J. Block and A. Dewald, “Windows memory forensics: Detecting (un)intentionally hidden injected code by examining page table entries,” Digital Investigation, vol. 29, pp. S3–S12, 2019.
16. K. D. Fairbanks, “An analysis of Ext4 for digital forensics,” Digital Investigation, vol. 9, pp. S118–S130, 2012.
17. K. H. Hansen and F. Toolan, “Decoding the APFS file system,” Digital Investigation, vol. 22, pp. 107–132, 2017.
18. G. B. Bell and R. Boddington, “Solid state drives: The beginning of the end for current practice in digital forensic recovery?” J. Digital Forensics, Security and Law, vol. 5, no. 3, pp. 1–20, 2010.
19. S. L. Garfinkel, “Carving contiguous and fragmented files with fast object validation,” Digital Investigation, vol. 4, pp. S2–S12, 2007.
20. S. Garfinkel, A. Nelson, D. White, and V. Roussev, “Using purpose-built functions and block hashes to enable small block and sub-file forensics,” Digital Investigation, vol. 7, pp. S13–S23, 2010.
21. J. Kornblum, “Identifying almost identical files using context triggered piecewise hashing,” Digital Investigation, vol. 3, pp. S91–S97, 2006.
22. V. Roussev, “Hashing and data fingerprinting in digital forensics,” IEEE Security & Privacy, vol. 7, no. 2, pp. 49–55, 2009.
23. D. Quick and K.-K. R. Choo, “Impacts of increasing volume of digital forensic data: A survey and future research challenges,” Digital Investigation, vol. 11, no. 4, pp. 273–294, 2014.
24. E. Casey, Digital Evidence and Computer Crime, 3rd ed. Waltham, MA, USA: Academic Press, 2011.
25. R. Harris, “Arriving at an anti-forensics consensus: Examining how to define and control the anti-forensics problem,” Digital Investigation, vol. 3, pp. S44–S49, 2006.
26. B. Carrier, “Defining digital forensic examination and analysis tools using abstraction layers,” Int. J. Digital Evidence, vol. 1, no. 4, 2003.