Forensic Artifacts Left by AI: Tool Capability Needs

A Practical White Paper for Digital Forensic Tool Developers

Author: Christopher L. T. Brown (clbrown@techpathways.com), Technology Consulting in Digital Forensics, InfoSec & AI, September 2026

Audience: Digital forensic investigators, tool developers, incident responders, and technical leaders evaluating next-generation forensic software capabilities.

Scope: Microsoft Windows and Apple macOS artifacts tied to local AI applications, browser services, autonomous coding agents, and AI data movement.

Purpose, Audience, and Scope

This white paper targets two core groups: forensic tool developers prioritizing their engineering roadmaps, and DFIR practitioners determining what evidence they can reliably extract today versus what still requires manual legwork.

Throughout this paper, "AI artifact" refers to a diverse family of evidence sources rather than a single file format. It encompasses local desktop applications, web-based interfaces, local model runtimes (such as Ollama, LM Studio, and llama.cpp), agentic coding CLIs, cloud audit logs, and standard operating system traces left behind by AI-driven processes.[1][11][20][31]

Because client applications, browser extensions, and cloud platforms evolve on rapid release cycles, every artifact description here is tied to specific versions and platforms. What holds true for one client on Windows may not apply to macOS or even to a minor update next month. Treat these findings accordingly and verify them in your test environments.[1][31]

Core Definitions: Artifact: Any recoverable trace of system or user activity. Indicator: An observable event suggesting activity without standalone proof. Telemetry: Structured operational event streams (e.g., Sysmon or audit logs). Log: A durable, append-only record of events. Cache/Database: Transient or semi-persistent local storage used by applications. Prompt: User- or agent-supplied input to an AI model. Completion: Text, code, or binary output produced by an AI model. Local: Stored directly on the examined endpoint. Cloud: Stored or managed within remote provider infrastructure.

1. Executive Summary

AI activity rarely vanishes without a trace. Instead, it scatters fragments across endpoints, browsers, cloud tenants, and local runtimes. An investigator trying to piece together an AI workflow cannot rely on a single database. They have to reconstruct the narrative across fragmented local caches, browser storage, cloud audit trails, volatile memory, and OS-level telemetry.[1][2][5][11][31]

The core engineering challenge for forensic tools is not simply capturing these raw files. It is normalizing, correlating, and interpreting them. Tools must connect prompts, completions, uploaded files, API transactions, session tokens, user accounts, and data-transfer spikes into a coherent, defensible timeline.[1][5][6][22]

On Windows, prioritize application execution evidence, browser storage, System Resource Usage Monitor (SRUM) network history, Jump Lists, LNK files, AmCache, ShimCache, Sysmon event streams, and AI-specific sources such as Microsoft Copilot audit records and Recall snapshots.[5][7][8][9][23][24][26]

On macOS, prioritize application containers, Application Support directories, browser profiles, FSEvents, Unified Logging, TCC permission databases, quarantine and provenance extended attributes, and QuickLook caches, while accounting for FileVault, Secure Token, and Apple Silicon acquisition constraints.[13][14][15][16][17][18][28]

Core Thesis: Digital forensic tools must approach AI activity as a multi-source reconstruction challenge. Effective tools identify AI-capable binaries, preserve volatile and persistent traces, correlate local artifacts with cloud audit logs, and present findings with clear confidence boundaries and attribution limits. Every step in the extraction pipeline must remain fully verifiable.

2. Key Findings and Recommendations

Table 1. Key findings and tool implications

Finding	Implication for forensic tools	Primary references
AI desktop and mobile applications can leave recoverable local artifacts such as cache, metadata, chat records, app state, and network traces.	Tools need application-specific parsers and generic fallback parsers for caches, SQLite stores, logs, and metadata.	[1][19]
Cloud and enterprise AI platforms may retain higher-authority audit and compliance records than endpoints, but access is licensing-, role-, and retention-dependent.	Tools need verifiable cloud ingest connectors and retention warnings, not just endpoint acquisition modules.	[2][3][5][6]
Browser-based AI usage inherits browser forensic artifacts: history, cookies, downloads, local storage, caches, sessions, and profile locations.	Tools should classify AI domains and reconstruct AI sessions from browser profiles across Windows and macOS.	[11][12]
Agentic AI and coding assistants can create command, file, and transcript traces that blur the line between user action and AI-mediated execution.	Tools must represent actor context: human user, local agent, remote agent, script, API key, service account, and workstation identity.	[20][21][25]
Provenance markers such as C2PA content credentials are useful but not dispositive because metadata can be absent, removed, or incomplete.	Tools should parse provenance but report it as corroborative evidence and preserve the original file and acquisition context.	[29][30]

- Build a unified AI artifact schema that captures event types, source paths, timestamp sources, actor/account IDs, hostnames, client applications, content availability, and confidence scores.
- Prioritize purpose-built parsers for web AI interfaces, ChatGPT/Copilot desktop clients, coding agents, Windows execution/network traces, and macOS container/log stores.
- Integrate cloud ingest connectors for the OpenAI Compliance Platform, OpenAI API audit logs, Microsoft Purview Copilot records, and Microsoft 365 eDiscovery exports.[2][3][5][6]
- Establish automated validation test suites with known prompts, uploaded files, downloads, and purge actions, allowing developers to verify exact parser recovery boundaries.[1][28]

3. Methodology and Assumptions

This paper synthesizes vendor technical specifications, DFIR research, operating system documentation, and real-world practitioner experience. It outlines concrete technical requirements for digital forensic software rather than evaluating specific commercial products.[1][5][7][13][14][28]

Examiners encounter three primary evidence states in the field: endpoint-only traces, combined endpoint-and-cloud evidence, and cloud-only records requiring administrative privileges or legal process.[2][3][5][6]

This analysis distinguishes between artifacts collectible with current tools (browser storage, OS execution artifacts, endpoint telemetry, application containers, and tenant log exports) and areas requiring active software development (session reconstruction, prompt-to-output linking, agent action attribution, and schema-resilient parsers).[1][5][11][12][20][24]

Depending on the deployment architecture, complete artifact recovery may demand live incident response, elevated privileges, memory acquisition, browser cache extraction, tenant administrative access, or formal legal subpoenas.[2][3][5][6][7][28]

Table 2. Evidence recovery states

State	Description	Tool implication
Collectible today	Artifact can be collected with existing forensic, endpoint, browser, cloud-export, or log-collection workflows.	Build parsers, workflows, correlation, and reporting around existing data sources.
Partially recoverable	Artifact exists only as metadata, remnants, cache, derived text, or provider-side references.	Report uncertainty and require corroboration from independent sources.
Requires new development	Artifact class is known but not consistently parsed, normalized, or correlated by mainstream tools.	Build version-aware parsers, schemas, and validation datasets.
Requires cloud-side process	Artifact is controlled by a provider or tenant service and may not exist on the endpoint.	Support API ingest, legal-process tracking, and retention warnings.

4. Windows Forensic Artifact Analysis

Investigating Windows AI activity follows the same logic as investigating traditional application misuse or data staging: identify what executed, which user account ran it, what files were accessed, where data traveled, and how cloud records corroborate the endpoint.[7][8][9][22][24]

Application execution artifacts remain foundational. AmCache retains executable metadata and SHA-1 hashes, while ShimCache records file paths and last-modified timestamps. When correlated with Prefetch and UserAssist, these sources confirm application presence and execution timelines.[8][9][10]

Where deployed, Sysmon captures granular process creation trees, parent-child process relationships, full command-line arguments, hashes, network connections, and file creation events. This logging is invaluable when analyzing autonomous agents that spawn background shells, script interpreters, or CLI utilities.[7]

SRUM provides essential network and resource tracking over time. It records cumulative byte transfers per application, helping investigators identify when an AI desktop client, browser, or background daemon uploaded or downloaded large datasets.[22][24]

LNK files and Jump Lists establish user interaction with specific documents and applications. They preserve target paths, volume serial numbers, and access timestamps, allowing examiners to prove that a user opened, edited, or moved an AI-generated file even if the file itself was later deleted.[23]

Browser-based AI interactions typically reside in Chromium or Edge user data directories. These yield browsing history, cached web responses, session cookies, local storage, IndexedDB databases, and file download entries.[11][12]

Microsoft Purview audit logs record enterprise Copilot and integrated AI transactions, capturing user identities, exact interaction timestamps, client IP addresses, and specific internal SharePoint or OneDrive resources referenced in model completions.[4][5][6]

On Copilot+ PCs, Microsoft Recall stores encrypted local snapshots of desktop activity. Because Recall snapshots are protected by Windows Hello authentication and isolated per user profile, forensic tools must treat Recall as a high-value but strictly access-controlled source that requires live authentication or authorized acquisition workflows.[26][27]

Table 3. Windows AI artifact matrix

Subcategory	Examples	Collection status	Interpretation notes
Installed AI tools and local components	ChatGPT Windows app, Copilot components, Ollama/LM Studio/llama.cpp, coding assistants, helper binaries	Collectible today; app-specific parsing needs development	Correlate app stores with Prefetch, AmCache, ShimCache, UserAssist/BAM, command-line auditing, and logs.[31][33]
Browser/web app artifacts	Chrome/Edge history, cookies, cache, local storage, IndexedDB, downloads	Collectible today	Useful for cloud AI sessions but may not contain full prompt/response content.
AI tool logs and telemetry	Claude/Codex session files, app logs, API client logs, Sysmon	Partially collectible today	Transcript files may be plaintext or structured JSONL depending on tool and version.[20][25]
File system/output handling	Generated files, Downloads, Recent items, LNK/Jump Lists, MFT/USN traces	Collectible today	Evidence of handling is not necessarily evidence of authorship or intent.
Network and exfil indicators	SRUM, browser downloads/uploads, endpoint telemetry, proxy/DNS logs	Collectible today if logs exist	Volume and timing require correlation with application and account context.

5. Apple macOS Forensic Artifact Analysis

Analyzing macOS systems requires evaluating distinct Darwin operating system mechanisms: sandboxed app containers, property lists, SQLite databases, FSEvents, Unified Logging, Spotlight metadata, TCC privacy permissions, and extended attributes.[13][14][17][18][28]

The Scientific Working Group on Digital Evidence (SWGDE) outlines specific acquisition standards for Intel and Apple Silicon Macs, noting that hardware encryption and modern security architectures dictate how physical and logical extractions must proceed.[28]

FileVault, Secure Token bindings, Bootstrap Tokens, and Apple Silicon APFS volume ownership directly determine whether examiners can access user data from powered-off machines or live sessions. Forensic software should record these volume states as part of its case metadata.[15][28]

The FSEvents daemon tracks filesystem changes across directories. This log is crucial for detecting when an AI tool, agent, or user created, staged, renamed, or purged prompt files, downloaded assets, or script directories.[13]

Apple Unified Logging consolidates subsystem activity in memory ring buffers and on-disk trace files. It provides precise visibility into application lifecycles and background daemons, though examiners must validate entries against known privacy redaction rules (such as `<private>` string masking).[14][18]

The Transparency, Consent, and Control (TCC) database manages application permissions for sensitive resources, including user folders, Full Disk Access, camera, microphone, and accessibility automation. Reviewing TCC grants helps establish how an AI client or agent accessed protected user directories.[17]

Extended attributes (specifically com.apple.quarantine and com.apple.provenance) indicate download sources, originating applications, and initial execution context for AI binaries and downloaded model weights.[16][18]

Browser-based AI usage on macOS concentrates in ~/Library/Application Support and ~/Library/Caches under Chromium, Safari, or Edge profile trees, preserving browsing history, WebSQL/IndexedDB records, and session data.[11][12]

Table 4. macOS AI artifact matrix

Subcategory	Examples	Collection status	Interpretation notes
Installed AI tools and local components	App bundles, app containers, Application Support, Caches, preferences, logs	Collectible today; app-specific parsing needs development	Version and sandbox behavior are critical.
Browser/web app artifacts	Chrome/Safari/Edge history, cache, cookies, local storage, downloads, sessions	Collectible today	Bridge endpoint activity to cloud AI services.
OS interaction traces	FSEvents, Unified Logs, recent items, QuickLook, Spotlight-related data	Collectible today with tooling limits	Often corroborative rather than content-bearing.
Permissions and privacy context	TCC.db, Full Disk Access, Apple Events, Downloads/Desktop/Documents access	Collectible with privileges	Permission grants help explain how an AI app or agent could access data.
Quarantine and provenance	com.apple.quarantine, com.apple.provenance	Collectible today	Useful for app/document origin analysis but not conclusive alone.

6. Cross-Platform Forensic Tool Requirements

The following specifications translate platform-specific artifact realities into concrete engineering requirements for forensic tool developers. Each capability separates the investigative problem from the required software architecture and outlines explicit evidentiary limits.

Table 5. Recommended forensic tool capabilities

AI Browser Session Reconstruction

Required field	Details
Investigator problem it solves	Investigators need to identify browser-based AI sessions and reconstruct prompts, uploads, downloads, and account context.
Why current tooling is insufficient	General browser tools often show visits but do not classify AI services, prompt-bearing storage, model-workspace context, or upload/download relationships.
Likely evidence sources / artifact locations	Chrome/Edge/Safari history, downloads, cookies, cache, local storage, IndexedDB, service-worker stores, browser sessions.
Technical description of proposed capability	Create AI-domain classifiers and parsers that normalize web-app session events into prompt, response, upload, download, account, and timestamp records.
Expected investigative value	High
Limitations / false-positive risks / attribution concerns	False positives from ordinary browsing; prompt text may be absent or encrypted; cloud-side records may be needed.
Priority level	High
Implementation difficulty	Medium
References	[11][12]

Desktop AI App Parser Framework

Required field	Details
Investigator problem it solves	Investigators need consistent extraction from rapidly changing local AI clients.
Why current tooling is insufficient	Current support is app-specific and often trails vendor release cycles.
Likely evidence sources / artifact locations	App containers, Application Support, AppData, caches, SQLite/JSON stores, logs, registry/plist files, crash reports.
Technical description of proposed capability	Build a version-aware parser framework that detects app identity, version, artifact schema, account state, cached content, and deletion remnants.
Expected investigative value	High
Limitations / false-positive risks / attribution concerns	App updates can break parsers; cached content may not equal cloud truth.
Priority level	High
Implementation difficulty	High
References	[1][19]

Agentic AI Transcript and Tool-Call Analyzer

Required field	Details
Investigator problem it solves	Agentic tools can read, write, run commands, and transfer files, but investigators need to know what the agent did versus what the user directly did.
Why current tooling is insufficient	Traditional timelines often collapse agent-mediated commands into the human user's account context.
Likely evidence sources / artifact locations	Claude Code ~/.claude, Codex sessions/history, shell history, terminal logs, editor logs, Sysmon, Unified Logs, file modification records.
Technical description of proposed capability	Parse agent transcripts and tool-call logs, correlate them with process/file/network events, and label actions as user-authored, agent-suggested, agent-executed, or externally triggered where evidence supports it.
Expected investigative value	High
Limitations / false-positive risks / attribution concerns	Transcript absence does not prove no agent activity; attribution may remain ambiguous.
Priority level	High
Implementation difficulty	Medium
References	[20][21][25][7]

Cloud AI Audit and Compliance Log Ingest

Required field	Details
Investigator problem it solves	Endpoint evidence may not contain the authoritative prompt, response, account, or file-reference record.
Why current tooling is insufficient	Many forensic tools do not ingest AI audit APIs or explain licensing and retention constraints.
Likely evidence sources / artifact locations	OpenAI Compliance Platform, OpenAI API audit logs, Microsoft Purview Copilot/AI audit records, eDiscovery exports, DLP/SIEM connectors.
Technical description of proposed capability	Add API connectors, export importers, retention warnings, chain-of-custody metadata, and correlation keys for account, time, source IP, resource references, and request IDs.
Expected investigative value	High
Limitations / false-positive risks / attribution concerns	Requires tenant rights, licensing, and sometimes legal process; retention windows vary.
Priority level	High
Implementation difficulty	Medium
References	[2][3][4][5][6]

AI Data-Staging and Exfiltration Correlator

Required field	Details
Investigator problem it solves	Investigators need to connect AI usage to file staging, upload, download, synchronization, and data movement.
Why current tooling is insufficient	Existing exfil workflows may not classify AI endpoints or correlate prompt/upload/output lifecycle events.
Likely evidence sources / artifact locations	SRUM, Sysmon, browser downloads/uploads, cloud audit logs, LNK/Jump Lists, MFT/USN, FSEvents, DNS/proxy logs.
Technical description of proposed capability	Build correlation views that show file touched -> AI tool opened -> upload/download/network volume -> generated output saved -> sync/export path.
Expected investigative value	High
Limitations / false-positive risks / attribution concerns	Network volume does not prove content; endpoints may be shared across legitimate services.
Priority level	High
Implementation difficulty	High
References	[22][24][23][13]

Prompt/Input Activity Evidence Model

Required field	Details
Investigator problem it solves	Investigators need to determine whether sensitive text or files may have been submitted to an AI system.
Why current tooling is insufficient	Tools lack a common way to represent typed prompts, pasted clipboard content, file pickers, drag/drop actions, screenshots, voice input, and API payloads.
Likely evidence sources / artifact locations	Clipboard artifacts where available, browser/app stores, local transcripts, audit logs, screenshots/Recall snapshots, keyboard/input telemetry if deployed.
Technical description of proposed capability	Create an evidence model that separates observed prompt text, inferred input, uploaded file references, and unobserved but possible input paths.
Expected investigative value	High
Limitations / false-positive risks / attribution concerns	Clipboard and input traces are often volatile or absent; do not overstate intent.
Priority level	High
Implementation difficulty	High
References	[1][5][26]

Generated Output Handling Parser

Required field	Details
Investigator problem it solves	Investigators need to track how AI output was saved, edited, exported, shared, or embedded in other documents.
Why current tooling is insufficient	Content-origin analysis is often separate from file-system timelines and document metadata review.
Likely evidence sources / artifact locations	Downloads, generated files, document metadata, QuickLook thumbnails, LNK/Jump Lists, FSEvents, cloud sync logs, C2PA metadata.
Technical description of proposed capability	Parse and correlate generated-output artifacts, including file creation paths, document metadata, provenance markers, thumbnail/cache evidence, and subsequent sharing/sync events.
Expected investigative value	Medium
Limitations / false-positive risks / attribution concerns	AI-generated content detection is unreliable by itself; provenance metadata can be absent or stripped.
Priority level	High
Implementation difficulty	Medium
References	[23][13][29][30]

Windows Recall and Snapshot Artifact Module

Required field	Details
Investigator problem it solves	Investigators may need to preserve and interpret local screen snapshots that include AI prompts, responses, files, or browser sessions.
Why current tooling is insufficient	Recall is new, opt-in, access-controlled, and security-architecture dependent; mainstream tools may not yet handle its stores or access model.
Likely evidence sources / artifact locations	Recall snapshots, OCR/search index, Windows Hello access context, Copilot+ PC policy settings, local protected storage.
Technical description of proposed capability	Detect Recall availability, acquisition prerequisites, policy state, snapshot time ranges, and searchable text; report access limitations and privacy controls.
Expected investigative value	Medium
Limitations / false-positive risks / attribution concerns	Not available on all systems; access may require live/authenticated collection and must respect legal authority.
Priority level	High
Implementation difficulty	High
References	[26][27]

macOS Permission and Provenance Correlator

Required field	Details
Investigator problem it solves	Investigators need to explain how an AI app or agent obtained access to protected files, automation, microphone, camera, or screen-related data.
Why current tooling is insufficient	macOS permission artifacts are often reviewed separately from app activity and file timelines.
Likely evidence sources / artifact locations	TCC.db, quarantine/provenance xattrs, Unified Logs, app entitlements, app containers, FSEvents.
Technical description of proposed	Correlate TCC grants, quarantine/provenance, app launch/log events, and file-system activity into a permission-aware

capability	timeline.
Expected investigative value	Medium
Limitations / false-positive risks / attribution concerns	Permissions show capability, not actual use; xattrs are version- and workflow-dependent.
Priority level	High
Implementation difficulty	Medium
References	[17][16][14][13]

Evidence Confidence and Caveat Reporter

Required field	Details
Investigator problem it solves	Investigators and courts need to distinguish observed artifacts from inferred conduct and unsupported hypotheses.
Why current tooling is insufficient	Current tools often list artifacts without explaining confidence, volatility, collection mode, or attribution limits.
Likely evidence sources / artifact locations	All parsed artifacts, validation metadata, acquisition logs, parser version matrix, known-answer test results.
Technical description of proposed capability	Add report fields for observed fact, inference, alternative explanations, collection mode, parser version, source volatility, and corroborating artifacts.
Expected investigative value	High
Limitations / false-positive risks / attribution concerns	Requires disciplined product UX and validation governance; may expose gaps that users expect tools to hide.
Priority level	High
Implementation difficulty	Medium
References	[1][28][29][30]

7. Engineering Implementation Roadmap

This roadmap balances immediate practitioner needs with long-term architectural stability. Phase 1 focuses on extracting evidence readily available in existing OS structures; subsequent phases introduce specialized parsers, cloud ingestion pipelines, and validation controls.

Table 6. Phased implementation roadmap

Phase	Build focus	Representative features	Rationale
Phase 1 — Foundation	Collect and normalize existing endpoint and browser evidence	Browser AI classifier; Windows execution/network/file handling correlation; macOS app/container/FSEvents/TCC ingest; source-confidence fields	Most evidence needed for initial AI investigations already exists in browser, OS, and app artifacts.[7][11][12][13][17][22][24]
Phase 2 — AI-specific coverage	Add common AI application and service parsers	ChatGPT/Copilot desktop modules; Claude/Codex transcript parsers; Microsoft Purview/OpenAI audit importers; generated-output parser	AI clients and enterprise services now expose distinct artifacts and APIs that should not be treated as generic web history.[1][2][3][5][6][20][25]
Phase 3 — Defensibility and scale	Add validation, version matrices, and reporting controls	Known-answer test suite; parser regression tests; artifact volatility scoring; caveat-driven report templates; provenance verification	AI artifacts change quickly, and tools must disclose coverage and limitations to support defensible findings.[1][28][29][30]
Product management priority: Build shared parsing and correlation infrastructure first. Avoid one-off AI parsers that cannot express actor, account, source, confidence, and acquisition-mode differences.			

8. Investigative Scenarios and Workflows

Table 7. Practical investigative scenarios

Scenario	Core question	Likely evidence	Collection urgency	Attribution caveat
Insider data disclosure through AI	Did a user submit sensitive documents or text to an AI service?	Browser/app artifacts, file access records, clipboard remnants if available, cloud audit logs, DLP/SIEM records, SRUM/network indicators	High: prompts, cache, sessions, and cloud logs may be volatile or retention-limited.[2][5][6]	Endpoint access by a user account is not always proof of human-authored input.
Browser-based AI use	Which AI service was used, under what account, and what files were uploaded or downloaded?	Browser history, cache, downloads, cookies, local storage, IndexedDB, session files, account artifacts	Medium to high: browser data can be cleared and synchronized state may change.[11][12]	History shows access, not necessarily prompt content.
Local AI desktop client	What local AI application ran, what it cached, and what files it created?	AppData/Application Support, caches, SQLite/JSON stores, logs, registry/plists, execution artifacts	Medium: local stores may persist but app updates can migrate or purge schemas.[1][19]	Cached content may be partial or stale.
Remote or agentic AI actions	Did an AI agent read, edit, move, upload, or execute files on the endpoint?	Agent transcripts, shell history, process trees, Sysmon, Unified Logs, file-system timelines, editor logs	High: command/transcript logs may be user-configurable and cleanup-prone.[20][25]	Agent-executed command under a user account may not equal user intent.
Generated output handling	Was AI-generated content saved, edited, shared, or exfiltrated?	Downloads, generated documents, metadata, C2PA manifests, LNK/Jump Lists, FSEvents, cloud sync/audit logs	Medium: output files may persist but provenance metadata can be removed.[23][29][30]	AI provenance markers are corroborative, not dispositive.

9. Limitations, Evidentiary Caveats, and Validation Standards

- Artifact presence depends heavily on application version, operating system build, local configuration, and active account state. Parsers should never claim universal coverage without maintaining a published, version-tested support matrix.[1][11][12][28]
- Endpoint artifacts rarely tell the entire story. Where available, cloud audit trails and tenant compliance logs provide authoritative timestamps, full prompt-response text, user IDs, and resource references that local caches may omit.[2][3][5][6]
- Cloud-side collection requires administrative rights, active licensing, proper retention settings, or legal process. In Microsoft Purview, non-Microsoft AI logging requires pay-as-you-go configuration and adheres to strict retention windows.[5]
- Local file deletion or cache clearing does not instantly wipe all forensic evidence, but recovered fragments should be treated as partial indicators requiring corroboration rather than complete records.[1][22][23]
- macOS FileVault, Secure Token status, and Apple Silicon ownership models dictate what data is accessible from cold storage versus live, authenticated disk images.[15][28]
- AI text classifiers and stylometric attribution tools are not legally defensible proofs of human authorship without independent corroboration. Similarly, C2PA provenance metadata helps establish file history but can be stripped or modified.[29][30]
- No individual artifact discussed in this paper proves malicious intent or direct user authorship on its own. Defensible findings require validated parsers, rigorous chain of custody, and multi-source corroboration across endpoint and cloud layers.
- Enterprise MDM profiles, browser security policies, and local DLP agents can redirect, mask, or encrypt forensic artifacts. The absence of a specific local artifact does not prove that an AI interaction never occurred.

Table 8. Tool validation checklist

Validation area	Minimum expectation
Known-answer images	Use controlled prompts, uploads, downloads, deletions, app versions, OS versions, accounts, and timestamps.
Parser version disclosure	Report parser version, supported artifact schema version, and last validation date.
Cross-source corroboration	Require at least two independent artifact classes for high-confidence conclusions where possible.
Acquisition-mode disclosure	State live, dead-box, cloud export, memory, browser-only, or tenant-audit collection mode.

Attribution language	Separate observed facts, tool-inferred events, examiner inference, and unsupported hypotheses.

10. Conclusion

AI forensic artifacts are real, probative, and scattered across endpoints, browser profiles, local runtimes, autonomous agent logs, and cloud tenants.[1][2][5][11][20]

The next generation of digital forensic software must help examiners answer three practical questions: what AI activity took place, what independent evidence substantiates that finding, and what confidence level should be assigned given known platform caveats and retention limits.

Tool developers who prioritize browser session reconstruction, desktop client parsing, coding agent transcript analysis, cloud audit ingestion, and explicit confidence reporting will deliver the defensibility that investigators and courts require.

Appendix A. Figure Specifications

Figure 1. Cross-Platform AI Artifact Evidence Map

Platform	User inputs	AI outputs	Session history	Local traces	Cloud traces	Metadata
Browser-based AI	Prompt text, pasted text, file uploads	Chat transcript, exported content	Recent chats, account history	Browser cache, cookies, local storage, downloads	Vendor logs, account activity, sync records	Timestamps, user ID, device ID, IP address
Windows desktop AI	Typed prompts, clipboard, drag/drop files	Response windows, exports, generated files	App history, recent sessions	App cache, temp files, registry, shortcuts	Service logs, subscription activity, sync	Logon times, machine name, account, time zone
macOS desktop AI	Prompt text, clipboard, file picks	Transcript panes, saved documents	Recent items, account session data	App support, cache, plist, downloads	Vendor logs, sync, account events	User profile, hostname, timestamps, network addresses
API / cloud AI	API requests, prompt payloads, uploaded content	Response payloads, generated files, callbacks	Request IDs, threads, job IDs	Client logs, scripts, environment files	Service logs, billing, token usage, audit logs	API key identity, source IP, request timestamps

Caption: Common artifact classes created by AI tools and primary evidence locations across Windows and macOS.

Figure 2. Capability Roadmap for AI Artifact Forensic Readiness

Discover	Collect	Analyze and Report	Outcome
Inventory AI tools, accounts, and workflows; identify artifact locations; map local, browser, mobile, and cloud sources	Preserve endpoint, browser, mobile, and cloud evidence; capture timestamps, hashes, and account context; reduce loss of volatile session data	Correlate prompts, outputs, metadata, and timelines; distinguish local artifacts from cloud-synced artifacts; present clear, repeatable conclusions	Repeatable, defensible AI artifact examinations

Caption: A phased engineering roadmap for building repeatable collection and analysis capabilities for AI artifacts.

References

[1] M. Wanniarachchi Kankanamge, N. McKenna, S. Carmona, S. M. Hasan, A. R. Shahid, and A. Imteaj, "Digital Forensic Investigation of the ChatGPT Windows Application," *arXiv preprint arXiv:2505.23938,* https://arxiv.org/html/2505.23938v1, May 29th, 2025, Last accessed September 18th 2026

[2] OpenAI, "OpenAI Compliance Platform for Enterprise and Edu Customers," *OpenAI Help Center,* https://help.openai.com/en/articles/9261474-openai-compliance-platform-for-enterprise-and-edu-customers, 2024, Last accessed September 18th 2026

[3] OpenAI, "Admin and Audit Logs API for the API Platform," *OpenAI Help Center,* https://help.openai.com/en/articles/9687866-admin-and-audit-logs-api-for-the-api-platform, 2024, Last accessed September 18th 2026

[4] Microsoft, "Audit Log Activities in Microsoft Purview," *Microsoft Learn,* https://learn.microsoft.com/en-us/purview/audit-log-activities, 2024, Last accessed September 18th 2026

[5] Microsoft, "Audit Logs for Copilot and AI Applications in Microsoft Purview," *Microsoft Learn,* https://learn.microsoft.com/en-us/purview/audit-copilot, 2024, Last accessed September 18th 2026

[6] Microsoft, "Use Microsoft Purview to Manage Data Security and Compliance for Microsoft 365 Copilot and Copilot Chat," *Microsoft Learn,* https://learn.microsoft.com/en-us/purview/ai-m365-copilot, 2024, Last accessed September 18th 2026

[7] M. Russinovich and T. Garnier, "System Monitor (Sysmon) v15.15," *Microsoft Sysinternals, Microsoft Learn,* https://learn.microsoft.com/en-us/sysinternals/downloads/sysmon, 2024, Last accessed September 18th 2026

[8] C. Souza, "AmCache Artifact: Forensic Value and a Tool for Data Extraction," *Securelist by Kaspersky,* https://securelist.com/amcache-forensic-artifact/117622, October 1st, 2025, Last accessed September 18th 2026

[9] P. Froklage, "ShimCache vs AmCache: Key Windows Forensic Artifacts," *Magnet Forensics,* https://www.magnetforensics.com/blog/shimcache-vs-amcache-key-windows-forensic-artifacts, October 25th, 2024, Last accessed September 18th 2026

[10] Psmths, "Windows Forensic Artifacts: Handbook of Windows Forensic Artifacts Across Multiple Windows Versions," *GitHub,* https://github.com/Psmths/windows-forensic-artifacts, 2024, Last accessed September 18th 2026

[11] The Chromium Projects, "User Data Directory Documentation," *Chromium Open Source Project,* https://chromium.googlesource.com/chromium/src/+/main/docs/user_data_dir.md, 2024, Last accessed September 18th 2026

[12] Foxton Forensics, "Google Chrome History Location and Forensic Artifacts," *Browser History Examiner Documentation,* https://www.foxtonforensics.com/browser-history-examiner/chrome-history-location, 2023, Last accessed September 18th 2026

[13] Apple Inc., "File System Events: Core Services Documentation," *Apple Developer Documentation,* https://developer.apple.com/documentation/coreservices/file_system_events, 2024, Last accessed September 18th 2026

[14] Apple Inc., "Unified Logging and Activity Tracing," *Apple Developer Documentation,* https://developer.apple.com/documentation/os/logging, 2024, Last accessed September 18th 2026

[15] Apple Inc., "Use Secure Token, Bootstrap Token, and Volume Ownership in Deployments," *Apple Platform Deployment Guide,* https://support.apple.com/guide/deployment/manage-secure-token-and-bootstrap-token-dep24dbdcf9e/web, 2024, Last accessed September 18th 2026

[16] H. Oakley, "Ventura Has Changed App Quarantine with a New xattr," *The Eclectic Light Company,* https://eclecticlight.co/2023/03/13/ventura-has-changed-app-quarantine-with-a-new-xattr, March 13th, 2023, Last accessed September 18th 2026

[17] C. Polop, "macOS Hardening and Security Protections: Transparency, Consent, and Control (TCC)," *HackTricks Wiki,* https://hacktricks.wiki/en/macos-hardening/macos-security-and-privilege-escalation/macos-security-protections/macos-tcc/index.html, 2024, Last accessed September 18th 2026

[18] J. Mussman, "Mac Forensics in 2026," *Jordan's DFIR Blog,* https://jmussman.net/posts/mac_dfir, February 9th, 2026, Last accessed September 18th 2026

[19] Oxygen Forensics, "How to Extract and Parse Data from ChatGPT," *Oxygen Forensics,* https://www.oxygenforensics.com/technical-resources/how-to-extract-and-parse-chatgpt-data, October 15th, 2025, Last accessed September 18th 2026

[20] Claude Developer Tools, "Where Are Claude Code Logs Stored?," *Claude Developer Documentation,* https://claude-dev.tools/docs/log-locations, 2026, Last accessed September 18th 2026

[21] Anthropic, "Claude Code Settings and Configuration Reference," *Claude Code Documentation,* https://code.claude.com/docs/en/settings-reference, 2026, Last accessed September 18th 2026

[22] P. Froklage, "Investigating Data Exfiltration: Key Digital Artifacts Across Windows, Linux, and macOS," *Magnet Forensics,* https://www.magnetforensics.com/blog/investigating-data-exfiltration-key-digital-artifacts-across-windows-linux-and-macos, October 9th, 2024, Last accessed September 18th 2026

[23] L. Jones, "Windows 10 Jump List and Link File Artifacts - Saved, Copied and Moved," *DFIR Review,* https://dfir.pubpub.org/pub/wfuxlu9v, September 2020, Last accessed September 18th 2026

[24] P. Froklage, "SRUM: Forensic Analysis of Windows System Resource Utilization Monitor," *Magnet Forensics,* https://www.magnetforensics.com/blog/srum-forensic-analysis-of-windows-system-resource-utilization-monitor, October 5th, 2022, Last accessed September 18th 2026

[25] OpenAI, "Codex and Agent Non-Interactive Execution Mode," *ChatGPT Learn Documentation,* https://learn.chatgpt.com/docs/non-interactive-mode, 2025, Last accessed September 18th 2026

[26] Microsoft, "Retrace Your Steps with Recall on Copilot+ PCs," *Microsoft Support,* https://support.microsoft.com/en-us/windows/ai/ai-features/retrace-your-steps-with-recall, 2024, Last accessed September 18th 2026

[27] D. Weston, "Update on Recall Security and Privacy Architecture," *Windows Experience Blog, Microsoft,* https://blogs.windows.com/windowsexperience/2024/09/27/update-on-recall-security-and-privacy-architecture, September 27th, 2024, Last accessed September 18th 2026

[28] Scientific Working Group on Digital Evidence (SWGDE), "SWGDE Best Practices for Apple macOS Forensic Acquisition (Version 1.0)," *SWGDE Standards & Guidelines, Document 23-F-005,* https://www.swgde.org/documents/published-complete-listing/23-f-005-swgde-best-practices-apple-macos-forensic-acquisition, March 15th, 2024, Last accessed September 18th 2026

[29] Coalition for Content Provenance and Authenticity (C2PA), "C2PA Technical Specification: Content Credentials Explainer (Version 2.4)," *C2PA Specifications,* https://spec.c2pa.org/specifications/specifications/2.4/explainer/Explainer.html, 2024, Last accessed September 18th 2026

[30] National Security Agency (NSA), Cybersecurity and Infrastructure Security Agency (CISA), and Department of Defense (DoD), "Content Credentials: Strengthening Multimedia Integrity in the Generative AI Era," *Cybersecurity Information Sheet (CSI),* https://media.defense.gov/2025/Jan/29/2003634788/-1/-1/1/CSI-CONTENT-CREDENTIALS.PDF, January 29th, 2025, Last accessed September 18th 2026

[31] S. Murtuza, "Forensic Implications of Localized AI: Artifact Analysis of Ollama, LM Studio, and llama.cpp," *arXiv preprint arXiv:2603.23996,* https://arxiv.org/html/2603.23996v1, March 25th, 2026, Last accessed September 18th 2026

[32] ForensicArtifacts Community, "Digital Forensics Artifact Repository: Machine-Readable Knowledge Base of Forensic Artifacts," *GitHub,* https://github.com/ForensicArtifacts/artifacts, 2025, Last accessed September 18th 2026

[33] Microsoft, "Command Line Process Auditing," *Microsoft Learn,* https://learn.microsoft.com/en-us/windows-server/identity/ad-ds/manage/component-updates/command-line-process-auditing, 2023, Last accessed September 18th 2026