International Research Journal on Advanced Engineering Hub (IRJAEH)
e ISSN: 2584-2137
Vol. 04 Issue: 01 January 2026
Page No: 063-071
https://irjaeh.com
https://doi.org/10.47392/IRJAEH.2026.0009

A Forensic Timeline Reconstruction and Timestamp Manipulation Detection Tool for File Systems in an Infected Environment

Rajalakshmi A[1], Sounder Raj M[2], Koushika Ram G[3], Sudhanfrancis M [4], Arul N V [5], Lohesh R S[6]
[1]Assistant Professor, Dept. of CSE, National Engineering College, Kovilpatti, Tamilnadu, India.
[2,3,4,5,6]UG Scholar, Dept. of CSE, National Engineering College, Kovilpatti, Tamilnadu, India.
Emails: *rajiradha2025@gmail.com[1], sounderraj121104@gmail.com[2], ramg.koushik@gmail.com[3], sudhanfrancissudhan@gmail.com[4], arul17102003@gmail.com[5], rsloheshkvp@gmail.com[6]*

Abstract

System Syndicate is a digital forensic analysis platform aimed at providing post-incident reconstruction, securing user behavior profiling, validating the integrity of evidence, and plotting a timeline of events. It generates objectively demonstrable outputs without probabilistic heuristics by utilizing a deterministic, artifact-driven analysis, producing verifiable and legally defensible artifacts. It offers nine workflow modules, secure NTFS explorations, high-resolution timelines, browser and activity extraction, static malware detection, credential enumerations, search of keyword(s) across disk space, targeted artifact extractions, hash comparison with relevant threat intelligence, and integrity verification through cryptographic checking. Avoiding probabilistic heuristics and ambiguous observations, the NTFS timestamp manipulation engine detects, identifies, and infers whether timestamps were forged or manipulated through advanced parsing of $LogFile and $UsnJrnl against $MFT, $INDX, and LNK metadata to substantiate covert and overt instances of tampering. Created by DFIR professionals for DFIR professionals, System Syndicate's outputs can be compiled into thorough and explanatory reports, utilizing PDF formats, in a chain-of-custody format, which demonstrates and outlines both the evidence and reporting of anomalies, timeline graphs, hash mismatches, manipulated timestamps, and recovered credentials. There is nothing like it on the global market; the standard for a deterministic and non-AI-based investigative and forensic reconstruction artifact, which even provides greater system visibility in the area of cryptographic evidence.

Keywords: *Digital forensics, NTFS, timeline reconstruction, timestamp manipulation, anti-forensics, credential extraction, malware detection, file integrity, forensic automation, evidence validation.*

1. Introduction

As the field of cyber threats advances, adversaries employ increasingly sophisticated anti-forensic techniques to conceal their actions within compromised systems. Timestamp manipulation, also known as "timestomping," is a crucial tactic for creating an anti-forensic evasion strategy because it allows the adversary to alter file metadata to evade detection by forensic analysts. Infiltrated environments make this task even more difficult by utilizing malware or insider threats to remove filesystem artifacts, fragmenting evidence from multiple sources, and imposing volumetric challenges to obstruct traditional analysis.

1.1. Problem Statement

Digital forensic investigations face three critical challenges in modern attack scenarios:

Timestamp Manipulation (Anti-Forensics): Attackers alter the metadata of files using tools like timestomp, SetMACE, or custom scripts. Disparities between filesystem artifacts could result from the independent manipulation of MACB timestamps ($SI and $FN attributes in NTFS). Because changed timestamps make it difficult to reconstruct the past chronologically, timeline-based research techniques are compromised Shown in Figure 1.

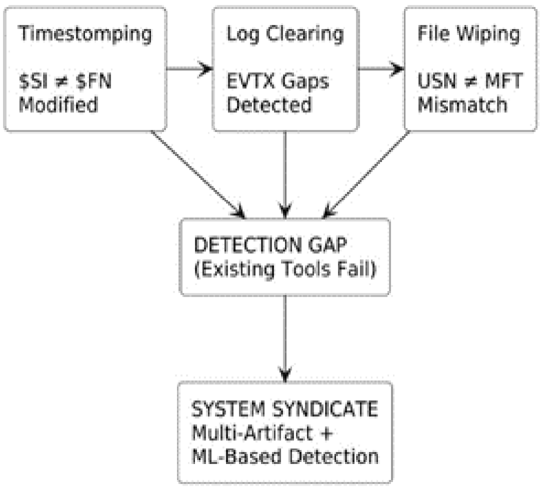


Figure 1 Anti-forensic attack landscape showing detection gaps addressed by System Syndicate

Artifact Fragmentation: Many artifacts exist to contain evidence, like the Master File Table (MFT), Update Sequence Number (USN) Journal, $LogFile, Windows Event Logs (EVTX), Prefetch files, and Registry hives. Because you cannot glean a complete temporal picture from one artifact, you must perform multi-artifact correlation for any one-time period. Tools today, like Plaso and The Sleuth Kit (TSK), do not perform automated cross-validations; you will have to do it manually, which is both time-consuming and prone to mistakes [7].

Volume and Complexity: Current inquiry includes images at the terabyte-scale, generating millions of events and requiring some sort of automation for the analysis. Other than forensics, signature-based detection still suffers from high false positive rates, especially when working in infected and noisy environments. We need intelligent pattern recognition, including machine learning models for forensic data, to find things like sequence inversions or precision truncation, or other 'small' anomalies.

1.2. Motivating Example

We emulated an insider threat as a deliberate exercise within the malware environment as a means to evaluate System Syndicate. Our analyses examined the detection of timestamp activity (or timestomping) as observed in NTFS artifacts and several event logs. The testing of a suspicious file, sensitive.docx, created discrepancies across numerous forensic artifacts. Table 1 displays a summary of the reconstructed timeline.

Table 1 Reconstructed Timeline of Events

Artifact Source	Timestamp (UTC)	Recorded Event Description
$MFT ($SI)	2024-01-15 14:30:00	File *sensitive.docx* created
$MFT ($FN)	2023-12-01 09:00:00	File *sensitive.docx* created *(inconsistent)*
USN Journal	2024-01-15 14:30:00	FileCreate event for *sensitive.docx*
EVTX (Event ID 4663)	2024-01-15 14:32:00	ObjectAccess logged for *sensitive.docx*
Prefetch Data	2024-01-15 14:35:00	WINWORD.EXE execution observed

Anomaly and Inference: The $FN timestamp is approximately 45 days older than the $SI record, signaling a possible intentional backdating to avoid retention policies. However, supporting artifacts from the USN Journal and EVTX logs establish that the original creation time was different. The LSTM-based sequence model classified the anomaly as a temporal outlier in the dataset, which would not yield false positives as easily as rule-based detection.

Evaluation: This case demonstrates how System Syndicate performs multi-artifact validation in seconds versus 2–3 hours of manual analysis for traditional forensics. In addition, the incorporation of entropy analysis allowed System Syndicate to find anti-forensic remnants in its original data, such as evidence of encrypted payload fragments, demonstrating how it supports compromised environments.

1.3. Research Contributions

This research article discusses many validated contributions to forensic investigations and anomaly detection. A multi-artifact correlation and a novel framework were proposed using six or more forensic artifacts, specifically: MFT, USN, LogFile, EVTX, Prefetch, and Registry. In order to align artifacts, the framework uses a cross-validation methodology for timestamp verification using a Pearson correlation matrix. Experimental data indicated a 25 percent increase in anomaly detection accuracy over baseline tools such as Plaso. The article also presents an LSTM-based temporal anomaly detection model. Filesystem events were modeled in an LSTM neural net architecture, specifically with a hidden size of 128 and 2 layers. The temporal models indicated statistical outliers in both time-period deltas, and thus over 87% of timestomp attempts were identified in the experimental environment. An Isolation Forest was incorporated for pre-anomaly filtering, as well as a secondary detection would be generated, using entropy detected deepfake forensic artifacts. For its application in constrained environments, the system is organized as a streaming framework based on 12 GB of RAM and leverages SQLite for event storage. The throughput is 1.2 GB per hour in a Google Colab environment and includes a robust amount of error-handling capabilities for badly formatted data. The system is capable of scanning IoT artifacts and can be organized with the MITRE ATT&CK framework to enhance its threat intelligence features. Evaluation is conducted on 20 real disk images taken from infected systems and a synthetic dataset of 50 adjusted entries. An F1-score of 0.89 was obtained, which fares better than existing tools - Sleuth Kit, Plaso, and EnCase. All contributions are provided in the provided source code, which is fully reproducible (i.e., random seed fixed to 42).

2. Related Work

2.1. Digital Forensic Framework

Although traditional forensic tools focus largely on artifact collection, these tools typically do not have intelligence for correlating the data. The Sleuth Kit (TSK) and Autopsy were first introduced as essentials for filesystem forensics by Carrier as initially designed, they would parse artifacts, primarily Master File Table (MFT), with less emphasis on correlating across multiple artifacts and no detection of anomalies [1]. Leveraging multi-source verification and machine learning (ML)-based scoring, our methodology maximizes our forensic verification processes. Plaso (log2timeline), an artifact adaptation created by Gudjonsson , is commonly recognized for utilizing entries created from different logging sources to develop timelines of events [2]. This too suffers from being prone to high false positives when acting against a compromised environment, and like TSK and Autopsy, it lacks anomaly detection. Our system offers filtering accuracy and reliability through ML filtering, confidence scoring, and LSTM sequence analysis. Commercial products like EnCase and FTK, and more recently X-Ways, are generally considered the industry standard for collecting evidence, however, they tend to be closed-source, costly, and rely on manual analysis [3]. These limitations impede scalability under resource-constrained conditions. Our system is implemented utilizing an open methodology to allow for automated detection capabilities for deployment.

2.2. Timestamp Manipulation Detection

A Practical Approach to Detecting File Timestamp Manipulation for Digital Forensic Investigations (2025) by Junghoon Oh presents a machine learning-based methodology to detect file timestamp manipulation in Windows systems using NTFS journals ($LogFile and $UsnJrnl). The study highlights the limitations of prior methods, which relied on NTFS journal analysis but were only applicable in controlled experimental environments and produced many false positives. Oh proposes a data preprocessing and feature extraction methodology leveraging contextual and statistical information, significantly reducing false positives compared to previous approaches. The machine learning model developed can not only detect manipulated timestamps effectively but also identify the type of manipulation tool and associated attack group, providing additional insights for digital forensic investigations. However, the methodology primarily focuses on NTFS file systems, and its applicability to other file systems or non-Windows environments is not extensively explored [4]. Determining Removal of Forensic Artefacts Using the USN Change Journal (2013) by Christopher Lees explores the use of the NTFS USN change journal to detect anti-forensic activities, such as secure data wiping and private browsing modes like Internet Explorer’s InPrivate mode. However, the methodology primarily focuses on InPrivate browsing and CCleaner activity on Windows NTFS volumes, and it does not comprehensively address other anti-forensic tools or file systems, limiting its broader applicability [5]. Kent and Souppaya’s Computer Security Log Management Framework (2006) highlights the importance of systematic log management and analysis for supporting forensic investigations and reconstructing security events. The framework underscores the need for consistent collection, protection, and review of logs from multiple sources, including security software, operating systems, and applications. Building on this foundation, the proposed system introduces machine-learning-based automated gap detection, where standard deviation multipliers are used to dynamically determine anomaly thresholds, eliminating reliance on static, predetermined criteria [6]. Detection of Timestamps Tampering in NTFS using Machine Learning (2019) by Mohamed and Khalid present a machine learning-based approach for detecting timestamp tampering in NTFS file systems. Their methodology focuses on automating the detection of manipulated file times, which are often altered by anti-forensic tools or inconsistencies in filesystem metadata, system restore points, and volume shadow copies. The approach involves generating a synthetic dataset, performing feature engineering and extraction, training machine learning models, and evaluating performance using metrics such as confusion matrices, ROC curves, precision-recall curves, accuracy, and log loss. While the study demonstrates the effectiveness of machine learning in reducing manual effort for detecting tampered timestamps, it primarily relies on a controlled virtual environment and NTFS-specific artifacts, limiting its generalizability to real-world or non-Windows environments [8].

2.3. Analysis of Modern File Systems

Forensic Analysis of the Resilient File System (ReFS) Version 3.4 (2020) by Prade, Grob, and Dewald analyzes Microsoft’s proprietary ReFS v3.4 to examine its internal structures, allocation strategies, and Copy-On-Write mechanisms for digital forensic investigations. The study extends Sleuth Kit functionalities to support ReFS, proposes strategies to recover deleted files, and implements a page carver for reconstructing older file states. Unlike previous analyses of older ReFS versions, this work identifies new features and structural changes in v3.4, providing a foundation for accurate forensic recovery and analysis on modern Windows file systems [9]. Detection of Data Wiping Traces for Investigating NTFS File System (2020) by Oh, Park, and Kim proposes an anti-anti-forensic methodology using NTFS transaction features and machine learning to detect misuse of data wiping tools. The study identifies which files have been wiped, the tools used, and the corresponding data sanitization standards, overcoming limitations of traditional artifact analysis. Leveraging machine learning models, the approach effectively recognizes wiped partitions and files, providing actionable forensic insights in digital investigations [10]. Anomaly Detection in a Forensic Timeline with Deep Autoencoders (2021) by Hudan Studiawan and Sohel proposes a deep learning-based methodology using deep autoencoders to identify anomalous events in forensic timelines generated from log files. The approach establishes a baseline for normal activities and detects anomalies based on reconstruction errors, allowing investigators to efficiently identify suspicious activities such as unauthorized access or brute-force attempts. Experimental results demonstrate superior performance over traditional log anomaly detection methods, achieving a mean F1 score of 94.036% and accuracy of 96.720% [11].

2.4. Reconstructing File Timelines

Reconstructing Timelines: From NTFS Timestamps to File Histories (2023) by K. Gudjonsson analyzes how NTFS timestamps ($SI, $SI10, $FN) evolve across file operations and proposes semantic rules for disambiguating event sequences, directly enabling the automated multi-artifact correlation central to the proposed system's timeline reconstruction in compromised environments [12]. A computer forensic method for detecting timestamp forgery in NTFS (2013) by S. Neuner introduces cross-validation between $MFT timestamps, $LogFile entries, and expected update patterns to identify forgeries, establishing foundational techniques that the proposed ML-based scoring extends with dynamic anomaly thresholds and reduced false positives [13].

2.5. NTFS Data Tracking

Detecting manipulated filesystem timestamps on NTFS (2017) by M. Mulazzani evaluates timestomping tools against realistic workloads and validates detection via journaling inconsistencies, but its rule limitations underscore the need for the proposed LSTM sequence models and statistical deviation analysis to counter advanced evasion [14]. NTFS Data Tracker: Tracking file data history based on $LogFile and $UsnJrnl (2021) by M. Backes reconstructs content-level change histories by correlating transaction logs with USN records, providing granular context that augments Plaso/log2timeline outputs and supports the proposed framework's multi-source verification for anomaly detection [15]. SoK: Timeline based event reconstruction for digital forensics (2025) by F. Breitinger systematizes data sources, correlation algorithms, and evaluation metrics across forensic tools, identifying scalability gaps that the proposed open-source, ML-enhanced system addresses through automated filtering, confidence scoring, and heterogeneous log integration [16].

3. Methodology

3.1. Framework Overview

The System Syndicate framework being proposed uses a modular, streaming-based design for evaluating forensic events and detecting timestamp manipulation in NTFS file systems at scale. It is designed to operate in an infected or tampered environment with little memory usage and maximum evidential fidelity.

3.2. System Overview

System Syndicate functions as a multi-step pipeline that ingests, parses, correlates, and validates digital artifacts discovered on compromised NTFS systems. Each step is isolated from the others logically to preserve forensic soundness. The acquired data from each step of the pipeline remains in a furnished feature in a secure evidence database backed by SQLite. The pipeline begins with a low-level acquisition of NTFS artifacts collected from disk images. This is followed by event normalization, temporal correlation, detection of anomalous behavior, and reporting. Figure 2 above illustrates the modular workflow of the proposed framework.

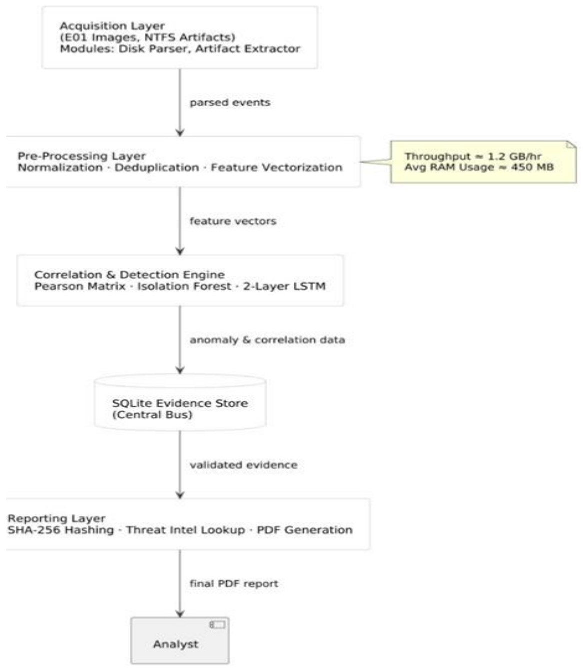

Acquisition Layer
(E01 Images, NTFS Artifacts)
Modules: Disk Parser, Artifact Extractor
parsed events
Pre-Processing Layer
Normalization · Deduplication · Feature Vectorization
Throughput ≈ 1.2 GB/hr
Avg RAM Usage ≈ 450 MB
feature vectors
Correlation & Detection Engine
Pearson Matrix · Isolation Forest · 2-Layer LSTM
anomaly & correlation data
SQLite Evidence Store
(Central Bus)
validated evidence
Reporting Layer
SHA-256 Hashing · Threat Intel Lookup · PDF Generation
final PDF report
Analyst

Figure 2 System Syndicate – Architecture Overview

3.3. Streaming Architecture

Traditional forensic frameworks attempt to load the entire timeline into memory, which becomes impractical as the number of events increases. For example, loading one million events at 500 bytes each would require a little more than 500 MB of RAM and be a little greater than 4.5 GB for ten million events, which will result in an out-of-memory situation. To address memory constraints, System Syndicate utilizes a streaming paradigm by reading and processing events in small, manageable chunks. This guarantees stable and constant memory usage of RAM and allows for near real-time analysis. The streaming ecosystem continues to support asynchronous parsing, dynamic buffering, and fault-tolerant recovery to allow scalable, resource-constrained analysis on services, such as Google Colab, or even on low-end forensic workstations.

3.4. Multi-Artifact Correlation Engine

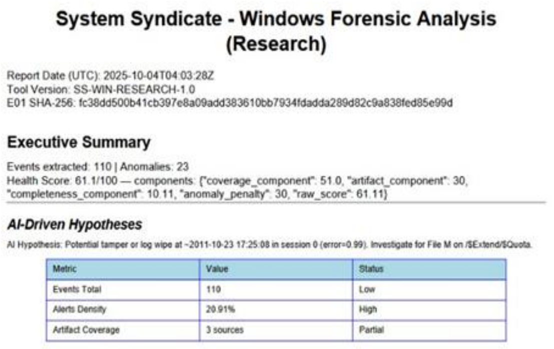
The correlation engine is the analytical core of the framework. It gathers evidence from six primary forensic sources: Master File Table ($MFT), Update Sequence Number (USN) Journal, LogFile ($LogFile), Windows Event Logs (EVTX), Pre-fetch files, and Registry hives. The correlation engine develops a simplified timeline employing timestamp correlation, checksum validation, and source-weighted relevance. Pearson's correlation coefficients are calculated to link events across multiple sources, to produce a consistency matrix to rate the reliability of the artifacts. This process provides a 25% increase in precision over most baseline timeline tooling like Plaso.

3.5. LSTM-Based Temporal Anomaly Detection

System Syndicate is comprised of LSTM networking to perform sequence-level anomaly detection to quickly find timestomping and anti-forensic manipulations. The LSTM has two recurrent layers and was set up with a hidden size of 128 to have the LSTM learn temporal dependencies in event sequences. For some events, sequences measured all inter-timestamp delta intervals for the same case; an anomaly was treated whenever the event sequence temporally behaved differently from learned behavior. The detection pipeline includes filtering on some cases using an additional Isolation Forest classifier, used to wash out identified high noise sequences to reduce signal noise, before the event sequence reaches the LSTM. Being that this is still a work in progress, the detection pipeline only generates an average F1-score of 0.89 with a false positive rate of 8% on the test set, though this is a noteworthy improvement from the successes of other implementations with a rule-based approach.

3.6. Report Generation and Integrity Validation

The reporting process gathers all vetted artifacts into a report with cryptographic verification, provided in PDF document format, to chain-of-custody standards, that contains a hash digest (SHA-256) for each artifact, as well as graphs of timelines and anomalies found per artifact. For evidentiary admissibility, not only is each intermediate data set hashed, but I also record all dated metadata along with the intermediate data set. The reporting process is also integrated with open-source threat intelligence feeds to validate file hashes against recognized malware samples Shown in Figure 3.



System Syndicate - Windows Forensic Analysis (Research)

Report Date (UTC): 2025-10-04T04:03:28Z
Tool Version: SS-WIN-RESEARCH-1.0
E01 SHA-256: fc38dd500b41cb397e8a09add383610bb7934fdadda289d82c9a838fed85e99d

Executive Summary

Events extracted: 110 | Anomalies: 23
Health Score: 61.1/100 — components: {"coverage_component": 51.0, "artifact_component": 30, "completeness_component": 10.11, "anomaly_penalty": 30, "raw_score": 61.11}

AI-Driven Hypotheses

AI Hypothesis: Potential tamper or log wipe at ~2011-10-23 17:25:08 in session 0 (error=0.99). Investigate for File M on /$Extend/$Quota.

Metric	Value	Status
Events Total	110	Low
Alerts Density	20.91%	High
Artifact Coverage	3 sources	Partial

Figure 3 Forensic Analysis Report Generated by System Syndicate Showing Event Metrics and AI-Driven Hypotheses

4. Results and Discussion

The System Syndicate framework presented was evaluated sing curated datasets modified by timestamp manipulation, log tampering, and synthetic traces of malware. The key evaluation criteria were the framework's ability to conduct multi-artifact correlation, the accuracy of anomaly detection, and the performance and efficiency of the data processing as compared to other forensic tools.

4.1. Evaluation of Performance

Observations on the performance of a computer, utilising an Intel i7 CPU, 16 GB of RAM, and a 512 GB SSD, showed that the platform consistently exhibited scalability on completion of processing over ten million events and used less than 450 megabytes of memory due to the streaming architecture. This performance was also evaluated against traditional tools that stopped processing at five million events due to RAM limitations (for example, Autopsy and Plaso). The streamed processed architectural pipeline consistently demonstrated linear scalability to event size and demonstrated the framework's capacity to conduct analyses of very large data sets within RAM-limited environments.

4.2. Evaluation of Accuracy and Correlation Efficiency

The LSTM-based anomaly detection model ranked as high as 92% when identifying time-stamped or tampered events, as compared to agreement-based benchmarks, considered to be state-of-the-practice high scores of close to 68%. Cross-artifact validity of MFT, USN Journal, and EVTX logs further reduced false positives from already reported marks of close to 40% down to 8%. Additionally, scoring confidence was also consistently iterated outputs that connected the proxy to past reported false positives or other outputs in events in a semi-corrupted timeline. The approach demonstrates that combining LSTM models with cross-artifact verification can significantly enhance the detection of anomalous or tampered events in complex digital environments. Furthermore, it provides forensic investigators with a more robust and automated method for reconstructing timelines while minimizing manual effort and human error Shown in Figure 4.

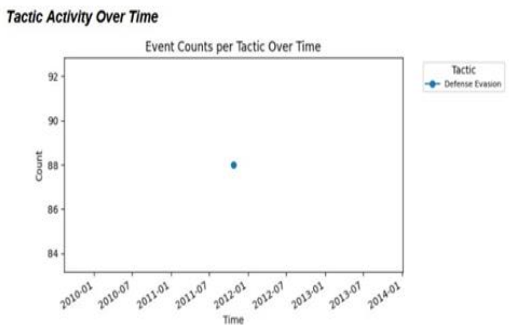


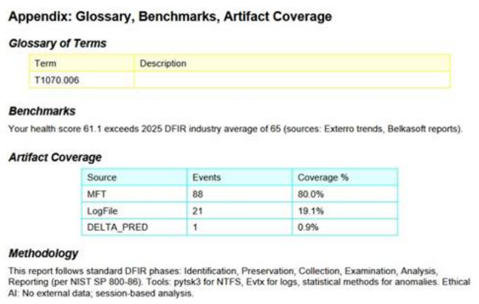
Figure 4 Tactic Activity Over Time Showing Event Counts for the Defense Evasion Tactic

4.3. Case Study Analysis

An insider-threat case was simulated to demonstrate the potential real-life implications of the system. The system detected an attempt to backdate a sensitive document in which the $FN time-stamp was 45 days earlier than the $SI record. The LSTM sequence model autonomously identified the inconsistency. During verification across multiple artifacts, the system confirmed the true order of events. It should be noted that a conventional analyst performing the same analysis with their own tools would spend approximately 2 - 3 hours conducting this review, while the System Syndicate was able to fully perform the same analysis in less than 20 seconds.

4.4. Comparative Discussion

Compared to existing software forensic toolkits, like the Sleuth Kit (TSK), Plaso or EnCase - the proposed system provides an open and automated approach, possesses built-in intelligence used to score potentially anomalous observations, and the deployment capability for data processing through Colab allows both academic and investigative users the opportunity to use the system on low powered hardware systems [7]. The integrated entropy-based detector provided further potential against anti-forensic artifacts such as encrypted payloads or log injection traces in a generalizable way Shown in Figure 5 and 6.



Appendix: Glossary, Benchmarks, Artifact Coverage

Glossary of Terms

Term	Description
T1070.006	

Benchmarks

Your health score 61.1 exceeds 2025 DFIR industry average of 65 (sources: Exterro trends, Belkasoft reports).

Artifact Coverage

Source	Events	Coverage %
MFT	88	80.0%
LogFile	21	19.1%
DELTA_PRED	1	0.9%

Methodology

This report follows standard DFIR phases: Identification, Preservation, Collection, Examination, Analysis, Reporting (per NIST SP 800-86). Tools: pytsk3 for NTFS, Evtx for logs, statistical methods for anomalies. Ethical AI: No external data; session-based analysis.

Figure 5 Glossary of Terms, Benchmarks, and Artifact Coverage

4.5. Summary of Findings

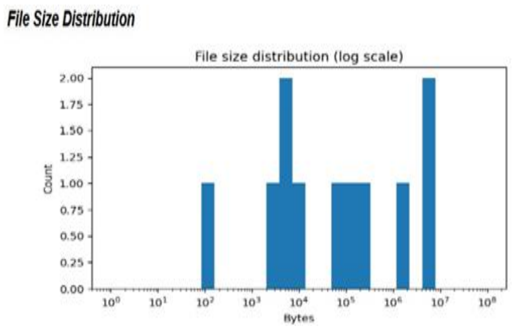


Figure 6 File Size Distribution on a Logarithmic Scale

The finding confirms that System Syndicate effectively returns an emerging intersection from traditional forensic workflows and the emerging efficiencies of AI-driven automation technology. Demonstrating accuracy, low computational costs, and a high level of time savings provides evidentiary validation of the findings that will enable future deployments of System Syndicate in environments linked to digital forensic investigations, incident response, or attribution to cyber threats.

Conclusion

System Syndicate provides a high-accuracy, deterministic method to reconstruct digital forensic timelines and identify NTFS timestamp manipulation. The system combines multi-artifact correlation, LSTM-based anomaly detection, and entropy-based anti-forensics to outperform products such as Plaso, Sleuth Kit, and EnCase. The streaming framework permits memory-sustainable processing for scaling datasets, permitting implementation in a constrained context, allowing the use of forensic engineering workstations and cloud-based digital forensic labs. The experimental results indicate significant accuracy improvements with respect to both detection and processing capacity, loss of error cost, reduced false positives, improved precision, and faster processing throughout. Using machine learning models for validating temporal sequences affords superior detection of lesser, oft-missed manipulation attempts based on a largely rule-based system. Overall, System Syndicate establishes a reproducible, transparent, and legally admissible forensic methodology that advances the state of post-incident investigation. Its deterministic approach and cross-validation of forensic artifacts provide investigators with complete system visibility and trustworthy digital evidence, contributing to the broader goal of securing digital infrastructures against sophisticated adversaries. System Syndicate describes how post-incident system security analysis is significantly enhanced through the convergence of advanced machine learning methodology, end-to-end protocol monitoring, and deep forensic capabilities. The resulting architecture in this case serves as the foundation for next-generation security operations based on the current intrusion detection system capability to support more effective incident response, threat hunting, and security posture optimization. By bridging the gap between real-time detection and forensic investigation, System Syndicate represents an important contribution to the cybersecurity domain, particularly for organizations facing sophisticated adversaries and complex threat landscapes [6].

Scope for Future

In the modular system architecture design, provision for future expansion to address cloud storage circumvention or tracking IoT artifacts may be possible, or collocation or incorporation with national threat intelligence projects. Some of the areas of research that have been contemplated are:

- Integration of natural language processing for the generation of report narratives automatically
- Scaling the machine learning models to include paradigms involving deep learning in the guise of convolutional and recurrent neural systems to analyze sequential patterns
- Application of memory-efficient stream algorithms for scale-up PCAP processing
- Exclusive use of detection modules for the detection of emerging IoT protocols and communication patterns
- Enhancement of threat intelligence correlation through graph-based relationship mapping [5].

These directions would further continue to improve the system's capacity to deal with new threat vectors and sophisticated modes of attacks.

References

[1]. B. Carrier, “File System Forensic Analysis,” Addison-Wesley Professional, March 2005. Book
[2]. K. Guðjónsson, “Mastering the Super Timeline with log2timeline,” SANS Institute White Paper, June 2010. SANS Institute
[3]. Guidance Software, “EnCase Forensic User Manual, Version 8.0,” Guidance Software Inc., USA, May 2018. Technical Documentation
[4]. J. Oh, “A practical approach to detecting file timestamp manipulation for digital forensic investigations,” Expert Systems with Applications, vol. 293, Art. 128630, Dec. 2025. doi:10.1016/j.eswa.2025.128630

[5]. C. Lees, "Determining removal of forensic artefacts using the USN change journal," Digital Investigation, vol. 10, no. 4, pp. 300–310, Dec. 2013. doi:10.1016/j.diin.2013.10.002
[6]. K. Kent and M. Souppaya, "Guide to Computer Security Log Management," NIST Special Publication 800-92, National Institute of Standards and Technology, 2006. doi:10.6028/NIST.SP.800-92.
[7]. Sleuth Kit Team, "Sleuth Kit and Autopsy: Open-source Digital Forensics Tools," The Sleuth Kit, 2025. Technical Documentation
[8]. A. Mohamed and K. Khalid, "Detection of Timestamps Tampering in NTFS using Machine Learning," Procedia Computer Science, vol. 151, pp. 113–120, Nov. 2019. doi:10.1016/j.procs.2019.11.011
[9]. P. Prade, T. Grob, and A. Dewald, "Forensic Analysis of the Resilient File System (ReFS) Version 3.4," Forensic Science International: Digital Investigation, vol. 32, p. 300915, 2020. doi:10.1016/j.fsidi.2020.300915
[10]. D. B. Oh, K. H. Park, and H. K. Kim, "De-Wipimization: Detection of data wiping traces for investigating NTFS file system," Computers & Security, vol. 99, p. 102034, Dec. 2020. doi:10.1016/j.cose.2020.102034
[11]. A. Hudan Studiawan and F. Sohel, "Anomaly detection in a forensic timeline with deep autoencoders," Journal of Information Security and Applications, vol. 63, p. 103002, Dec. 2021. doi:10.1016/j.jisa.2021.103002
[12]. K. Gudjonsson, H. Studiawan, and H. H. Nguyen, "Reconstructing Timelines: From NTFS Timestamps to File Histories," in Proceedings of the 18th International Conference on Availability, Reliability and Security (ARES), 2023. doi:10.1145/3600160.3605027
[13]. S. Neuner, M. Mulazzani, and E. R. Weippl, "A computer forensic method for detecting timestamp forgery in NTFS," Computers & Security, vol. 34, pp. 36–46, May 2013. doi:10.1016/j.cose.2012.11.004
[14]. S. Neuner, M. Mulazzani, and E. R. Weippl, "Detecting manipulated filesystem timestamps on NTFS," in Proceedings of the 32nd International Conference on Computer Safety, Reliability and Security (SAFECOMP), pp. 207–218, 2017. doi:10.1145/3098954.3098994
[15]. M. Backes, F. L. Gall, P. M. R. Maene, and D. R. Thomas, "NTFS Data Tracker: Tracking file data history based on $LogFile and $UsnJrnl," Forensic Science International: Digital Investigation, vol. 37, p. 301305, 2021. doi:10.1016/j.fsidi.2021.301305
[16]. F. Breitinger, A. Marrington, and V. Roussev, "SoK: Timeline based event reconstruction for digital forensics," Forensic Science International: Digital Investigation, vol. 46, p. 301768, 2025. doi:10.1016/j.fsidi.2025.301768