International Journal of Recent Engineering Research and Development (IJRERD)
ISSN: 2455-8761
www.ijrerd.com || Volume 11 – Issue 03 || May - Jun 2026 || PP. 28-36

A Novel Memory Forensics Methodology for Recovering File System Activities via Volatile Memory Analysis in Windows OS Systems

Dija S[1]
[1]*Centre for Development of Advanced Computing(C-DAC), Thiruvananthapuram, India*

Abstract: Traditional disk forensics approaches are increasingly insufficient against adversarial techniques such as full-disk encryption, secure deletion, and anti-forensic tooling. In live system analysis scenarios, volatile artefacts often constitute the primary source of evidence, making memory forensics the dominant investigative vector. This paper introduces a novel methodology for recovering major file system activities through the forensic analysis of volatile memory in Windows operating systems. The methodology encompasses a structured memory acquisition process, followed by artefact analysis that leverages GUID-based memory structures unique to the Windows OS environment. Experimental evaluation demonstrates the capability of the proposed approach to recover the file name, folder holding the file, and the program used to access the file. Results confirm that volatile memory analysis constitutes an indispensable and complementary evidence source in Windows digital investigations, capable of recovering artefacts beyond the reach of traditional forensic methodologies. Furthermore, a Windows system's hard disk may contain millions of files, making comprehensive forensic analysis extremely challenging. The recovery of file system activity related artefacts from volatile memory through the proposed methodology provides investigators with a focused and actionable list of recently accessed or modified files. This effectively narrows the investigative scope from an overwhelming volume of data to a manageable and forensically significant subset, substantially improving investigation efficiency, reducing analysis time, and enabling more targeted and precise forensic examination.
Keywords: Digital Forensics, Windows Forensics, Memory Forensics, Volatile Memory Forensics.

1. Introduction

Digital forensics encompasses the identification, acquisition, preservation, examination, and presentation of digital evidence in a manner that is admissible before a court of law. This procedure is critical in any Cyber Crime Investigation. Traditionally, digital forensic investigations relied heavily on the examination of persistent storage media, with investigators acquiring bit-stream images of storage devices at the scene of crime and subsequently analyzing those images at the laboratory recover files, metadata, and operating system artefacts.

The rapid evolution of computing technologies, operating systems, and anti-forensic techniques has introduced significant and growing challenges to this traditional investigative model. The nature and availability of digital evidence vary considerably depending on the operating system, its version, the applications installed, and the specific activities performed on the system. Consequently, investigators must continuously adapt forensic methodologies to identify and recover emerging forms of evidence from modern computing environments. Traditional disk forensics approaches are increasingly insufficient against adversarial techniques such as full-disk encryption, secure deletion, and anti-forensic techniques. Full-volume encryption technologies such as BitLocker render stored data inaccessible without cryptographic keys that may exist only transiently in volatile memory. Anti-forensic utilities capable of secure deletion, timestamp manipulation, and log sanitization actively eliminate disk-based evidentiary traces. Fileless malware executes entirely within memory using legitimate operating system facilities, leaving no executable artifacts on persistent storage. In live system analysis scenarios — where the suspect machine is encountered in a powered-on state — volatile artifacts frequently constitute the primary, and in many cases the exclusive, evidentiary source. These developments collectively position memory forensics as the dominant investigative vector in contemporary digital investigations [1].

Depending on the operational state of a system at the time of seizure, digital forensic investigations are generally classified into two approaches: Live Forensics and Offline Forensics. When a system is encountered in a powered-on state, investigators have the opportunity to acquire volatile memory before they are permanently lost upon shutdown. Conversely, when a system is found powered off, the investigation is necessarily limited to evidence residing on persistent storage. Both approaches carry distinct evidentiary strengths and limitations that must be understood in the context of the specific investigation.

Although storage-based forensic analysis remains an essential investigative approach, many of the most valuable artifacts generated during system operation exist only in volatile memory. User interactions with files and directories, process execution, network communications, and other operating system activities frequently leave traces in physical memory that are never fully preserved on persistent storage. In such environments, physical memory becomes a uniquely rich repository of evidentiary material, containing active and recently terminated process records, open file handles, network connection states, cryptographic key material, clipboard contents, and kernel data structures that collectively describe the complete operational context of the system at the moment of acquisition.

A further practical challenge of storage-based investigation is scale: a non-volatile storage media like disk may contain millions of files, making the identification of forensically relevant files a time-consuming and resource-intensive undertaking. The recovery of file system activity artifacts directly from volatile memory can address this challenge by providing investigators with a focused and actionable list of recently accessed or modified files — effectively narrowing the investigative scope for first level analysis from an overwhelming volume of data to a manageable and forensically significant subset. This capability substantially improves investigation efficiency, reduces analysis time, and enables more targeted and precise forensic examination. This paper introduces a novel methodology for recovering major file system activities through the forensic analysis of volatile memory in Windows operating systems. The first step in this methodology is the acquisition of Memory dump, followed by analysis that leverages GUID-based memory structures unique to the Windows environment. Experimental evaluation demonstrates the capability of the proposed approach to recover the file name, the folder holding the file, and the program used to access the file — artifacts that collectively enable precise reconstruction of user interactions with the file system. Results confirm that volatile memory analysis constitutes an indispensable and complementary evidence source in Windows System's digital investigations, capable of recovering artifacts beyond the reach of traditional forensic methodologies and substantially enhancing investigative efficiency. The remainder of this paper is organized as follows: Section 2 reviews the foundational distinction between live and offline forensic approaches; Section 3 details the proposed methodology; Section 4 presents experimental evaluation and results; Section 5 mentions the challenges and Section 6 concludes the research outcome.

2. Live and Offline Forensics

2.1 Live Forensics

Live forensics refers to the acquisition and examination of volatile data from a system that remains in an operational state at the time of the forensic response. The primary artifact targeted in live forensics is the contents of physical memory (RAM), which holds a transient but forensically rich record of system state at the moment of acquisition. This volatile data is irrecoverably destroyed upon system shutdown, making timely acquisition imperative when the suspect machine is found powered on.

The forensic value of physical memory is substantial and multifaceted. A RAM acquisition can yield active and recently terminated process listings along with their associated metadata; open file handles and memory-mapped file regions; network socket states and active connection tables; loaded kernel modules and injected code segments; dynamic-link library (DLL) mappings within process address spaces [2]; command-line arguments passed to processes at execution time; clipboard contents and remnants of recently executed activities; plaintext credentials and cryptographic key material that would otherwise be inaccessible in encrypted form on disk; and Windows Registry hive data cached in memory, including keys that may not yet have been flushed to persistent storage. These artefacts are crucial in any Cyber Crime Investigation as it provides the details of recently completed or ongoing activities in a system, which is often unattainable through storage-based analysis alone [3].

2.2 Offline Forensics

Offline forensics is the traditional Digital Forensics Methodology, applied to systems that are found powered off or that have been shut down prior to forensic response. In this methodology, the investigator creates a forensically sound, bit-stream copy of the storage media using write-blocking hardware to prevent any modification of the original evidence. This image is analyzed in a controlled laboratory environment, ensuring the integrity of the original media is maintained throughout the investigation.

The primary analytical focus of offline forensics is the file system. Through file system–level parsing, investigators can reconstruct all the file and folders the disk including the deleted files which are not yet overwritten. Offline analysis encompasses a range of techniques including: keyword and file searching; hash-based file identification against known good and known bad artifact databases; timeline analysis correlating file system timestamps — Modified, Accessed, Created, and Entry Modified (MACE) — with system event logs; signature mismatch analysis to detect file types concealed by false extensions; email analysis; application-

specific artifact examination; and the analysis of OS-specific artifacts such as the Windows Registry, prefetch files, link files (LNK), jump lists, Recycle Bin metadata, Windows Event Logs, and browser artifacts.

Despite its maturity and broad legal acceptance, offline forensics carries inherent evidentiary limitations in the context of modern threats. Encryption of full volumes through technologies such as BitLocker, or of individual files, renders disk contents inaccessible without the corresponding cryptographic keys. Anti-forensic utilities capable of secure deletion, timestamp manipulation, and log sanitization further diminish the evidentiary yield from persistent storage. Critically, activities that exist solely in volatile memory, or that are removed before being committed to disk, are entirely irrecoverable through traditional offline forensic methods [4]. Moreover, the sheer scale of modern storage environments renders exhaustive manual examination impractical and necessitates more targeted investigative approaches.

The complementary nature of live and offline forensics is therefore well established: the integration of memory forensics with conventional disk-based analysis provides a more comprehensive understanding of system activities and user behavior than either approach can achieve independently. A complete forensic investigation ideally incorporates both modalities, with physical memory acquisition taking precedence when the system is encountered in a running state. The methodology proposed in this paper addresses precisely this intersection — demonstrating how file system activity artifacts, traditionally the exclusive province of offline disk analysis, can be recovered and reconstructed directly from Windows volatile memory through the analysis of GUID-based searching, providing investigators with a focused, actionable, and forensically significant evidentiary capability that complements and substantially enhances traditional storage-based forensic examination.

3. Proposed Memory Forensics Methodology

3.1 Acquisition of Physical Memory Content from Windows OS Systems

The acquisition of physical memory constitutes the foundational step of any memory forensics investigation. Physical memory acquisition involves capturing the complete contents of a system's RAM to a binary file — commonly referred to as a memory dump or memory image — which is subsequently analyzed for offline examination. Memory acquisition can be accomplished through hardware-based or software-based approaches. Hardware-based methods, such as Direct Memory Access (DMA) acquisition via FireWire, Thunderbolt, or dedicated PCIe acquisition devices, operate independently of the target OS and are particularly valuable when kernel-level compromise is suspected. However, software-based acquisition tools are the most widely adopted approach in operational investigations due to their practicality and ease of field deployment. Widely used tools for Windows memory acquisition include DumpIt, WinPmem, and Magnet RAM Capture [5]. For the purposes of this research, physical memory acquisition was performed using DumpIt, a lightweight and forensically accepted utility that acquires a complete raw image of physical memory with minimal system interaction, as illustrated in Figure 1.

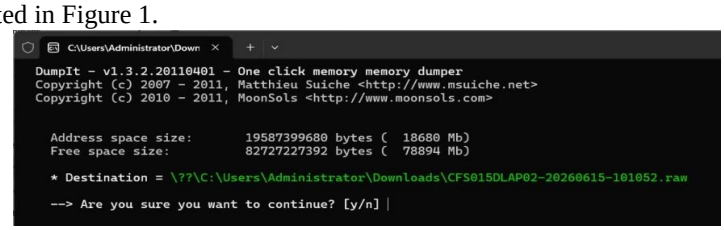

Figure 1: Physical Memory Dump Collection using DumpIt Tool

3.2 Analysis of the Memory dump to identify File System Activities.

Physical memory content analysis constitutes the most technically demanding phase of the memory forensics investigation, wherein the acquired memory image is systematically examined to identify, extract, and interpret forensically significant artifacts. Unlike file system analysis, which operates on structured, well-documented storage formats, memory analysis requires the examination of a large, largely unstructured binary image that interleaves kernel data structures, process address spaces, cached file system data, and transient runtime information across its entire extent. The memory dump collected from the suspect machine is analyzed by reading its content sequentially and structurally, targeting specific patterns, signatures, and data structures to extract evidence relevant to the investigation.

Analysis of a Windows physical memory image can yield artifacts spanning multiple investigative categories: process-level activities including running and recently terminated processes, loaded modules, and injected code [6]; file system activities reflecting user interactions with files and directories; network-related activities including active and recently closed socket connections and communication buffers; and browser-related activities capturing details of recently visited websites, search queries, and downloaded content[7]. Dija and Fernandez demonstrated the recovery of a wide range of forensic artefacts from RAM images acquired from

different Windows operating system versions, highlighting the evidential significance of volatile memory in modern digital investigations [8].This breadth of recoverable evidence makes physical memory analysis a uniquely powerful investigative technique, capable of reconstructing user activity with a temporal resolution and completeness that storage-based analysis frequently cannot match[9].

A central practical challenge in physical memory analysis is the identification of specific artifact categories within an image that may span several gigabytes in size. Manual inspection of a raw memory dump is operationally infeasible for the recovery of file system activity artifacts, given the volume of data involved and the absence of high-level structural organization. Automated, pattern-based identification of relevant memory regions is therefore essential [10]. In-depth research conducted as part of this work into the recovery of recently accessed file records from Windows memory images revealed a consistent and exploitable pattern: the physical memory of Windows operating systems stores recently accessed file names in association with a specific Globally Unique Identifier (GUID) value that precedes the file path records within the memory image [11]. This GUID value functions as a reliable and consistent binary signature that can be used to locate file system activity records within the memory dump through automated search procedures.

The GUID value leveraged by the proposed methodology is obtained from the ActivitiesCache.db file, a SQLite database maintained by the Windows OS to record application and file activity for the Timeline feature introduced in Windows 10. This database is located on the system at the path C:\Users\<username>\AppData\Local\ConnectedDevicesPlatform\<UserID>\ActivitiesCache.db. Examination of this database using a SQLite viewer reveals a table named Activity, within which the field AppActivityId contains the GUID value associated with file system access events, as illustrated in Figures 2. Critically, this GUID value is constant across the majority of Windows 10 minor versions, establishing it as a stable and version-independent forensic signature applicable across the broad installed base of Windows 10 deployments encountered in operational investigations. However, on some Windows 10 systems, a different GUID value is observed. The corresponding GUID can be readily extracted from the ActivitiesCache.db file of that system.

The novel methodology proposed in this paper proceeds through a structured two-phase analytical process. In the first phase, the AppActivityId GUID value is extracted from the ActivitiesCache.db file on the suspect system or its forensic image, establishing the binary signature to be used in memory search operations. In the second phase, the acquired physical memory image is searched for occurrences of this GUID value. Each occurrence within the memory dump identifies a memory region associated with a file system access event, immediately following which the file path of the recently accessed file is stored in memory. Extraction of these file path records yields the complete file name, the directory path identifying the folder containing the file, and — where present — the name of the application used to access the file. This three-component artifact set provides investigators with precise, actionable intelligence regarding recent file system interactions on the suspect system.

The methodology has demonstrated the capability to recover file path records across a wide range of file types, including document formats such as .txt, .docx, and .pdf; multimedia files including .mp4; and presentation files such as .pptx. Recovered records include not only the file name but also the complete directory path, enabling investigators to determine the exact location of the file within the storage hierarchy — whether on an internal hard disk, a solid-state drive, or removable storage media. This directory path information carries significant forensic value beyond mere file identification: even in cases where the file has been subsequently deleted from persistent storage, the presence of its full path record within volatile memory constitutes direct evidence that the file existed on the system and was accessed during the period captured by the memory image. This capability is particularly relevant in investigations involving the deletion of crime-related files, where memory-resident path records may provide the only surviving evidence of a file's prior existence and access on the suspect machine.

Furthermore, since a Windows system disk may contain millions of files, the recovery of recently accessed file records through the proposed methodology provides investigators with a highly focused and immediately actionable subset of the total file population. Rather than conducting exhaustive keyword searches or timeline analysis across an entire storage volume, the investigator is presented with a concise list of files that were demonstrably accessed in the period preceding memory acquisition — substantially narrowing the investigative scope, reducing analysis time, and enabling more precise and efficient forensic examination. This efficiency gain is of particular operational significance in time-sensitive investigations and in cases involving large-capacity storage media where comprehensive manual examination would otherwise be impractical.

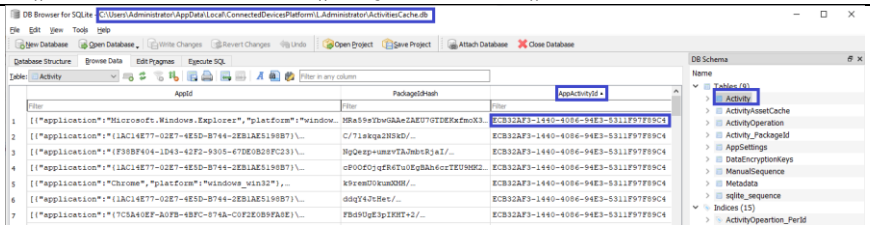

Figure 2: AppActivityId field in the Activity table of ActivitiesCache.db containing the GUID value used as a pivot artifact for memory-based evidence recovery.

4. Experiments and Results

To validate the proposed methodology, a series of experiments were conducted on Windows systems running different versions of the Windows 10 operating system. For each experiment, a physical memory dump was acquired from the target system after opening and accessing various file types. The acquired memory images were subsequently analysed using WinHex, Hexadecimal Viewer, to identify memory artefacts associated with file-system activities. The proposed methodology begins with extracting the AppActivityId value from the Activity table of the ActivitiesCache.db database available on the target system. The selection of ActivitiesCache.db as the starting point for the investigation is supported by earlier research demonstrating the forensic value of Windows Timeline artefacts in reconstructing user activities and application usage patterns [12] in latest Windows 10 releases. During the in-depth research and experiments, the AppActivityId value **"ECB32AF3-1440-4086-94E3-5311F97F89C4"** was identified and used as the primary search key for memory analysis. The memory dump was opened in WinHex, and a keyword search was performed using the GUID value identified from the ActivitiesCache.db file. The algorithm for the novel methodology for the GUID-Based File System Activity Recovery from Windows Volatile Memory is given below.

Step 1: Start
Step 2: Locate ActivitiesCache.db in the path:
OS-Drive:\Users\<username>\AppData\Local\ConnectedDevicesPlatform\<UserID>\ActivitiesCache.db
Step 3: View ActivitiesCache.db as a SQLite database using any free DB Viewer
Step 4: Query table [Activity], field [AppActivityId] and extract GUID value, G-Val; IF no G-Val is found → GO TO Step 9
Step 5: Load the memory dump collected from the suspect's system, MemDump.raw, into any Hexadecimal Viewer
Step 6: Search MemDump.raw for all occurrences of G-Val
Step 7: FOR EACH search hit H DO
Extract FileName, Folder Name, and Application (as highlighted in Figures 3–9)
Add the entry to Table T
Step 8: Output Table T as a structured file system activity report recovered from the Memory dump.
Step 9: Stop

The search results consistently revealed file-system-related artefacts stored in memory regions adjacent to the identified AppActivityId. Figures 3–9 present screenshots obtained from WinHex showing the results of the memory analysis. Each screenshot demonstrates the successful recovery of file-system activity information corresponding to a different file type. The recovered artefacts include HTML files (.html), text files (.txt), Microsoft Word documents (.docx), Microsoft PowerPoint presentations (.pptx), Portable Document Format files (.pdf), and multimedia files (.mp4). Table 1 presents the filenames recovered from the memory dump along with their associated forensic artefacts.

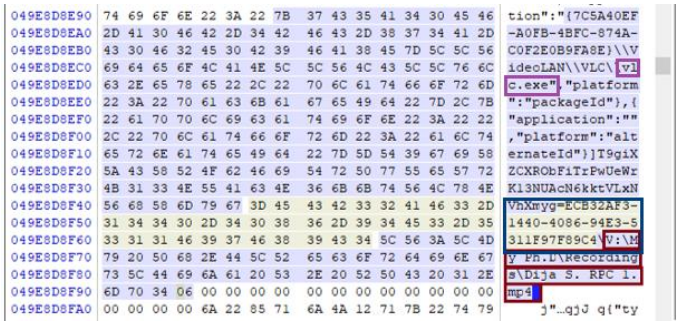

Figure 3: AppActivityId GUID occurrence in the memory dump associated with an HTML file (.html).

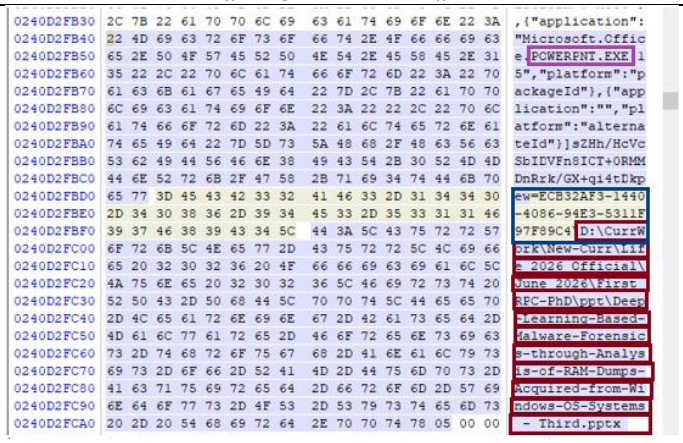

Figure 4: AppActivityId GUID occurrence in the memory dump associated with a text file (.txt).

A significant observation from the recovered memory artefacts is that the filename is not stored in isolation. Instead, the complete file path, including the directory structure leading to the file, is preserved within the memory region. Furthermore, the application associated with the file is also present in the recovered artefact. Consequently, the methodology provides three important pieces of forensic information simultaneously: (i) the filename, (ii) the exact file location within the storage device, and (iii) the application used to access the file.

Table 1: The type of files retrieved from the memory dump

Sl. No	Figure	File Type	File Path Recovered	Application Recovered
1	Figure.	.html	Yes	Browser
2	Figure.	.txt	Yes	Notepad
3	Figure.	.docx	Yes	MS Word
4	Figure.	.pptx	Yes	MS PowerPoint
5	Figure.	.pdf	Yes	PDF Reader
6	Figure.	.mp4	Yes	Media Player

The experimental results demonstrate that searching for the AppActivityId value within the memory dump provides a reliable mechanism for locating recently accessed file-system artefacts. In many instances, complete path information was recovered, enabling investigators to determine the exact location of the file within the file system hierarchy. Such information can significantly assist forensic investigators in identifying files relevant to an investigation without performing exhaustive searches across the entire storage medium.

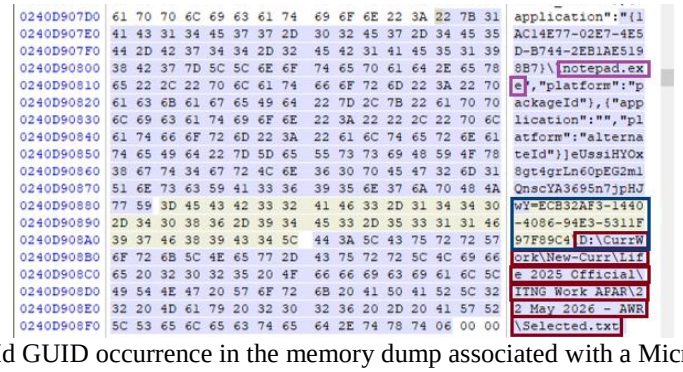

Figure 5: AppActivityId GUID occurrence in the memory dump associated with a Microsoft Word document.

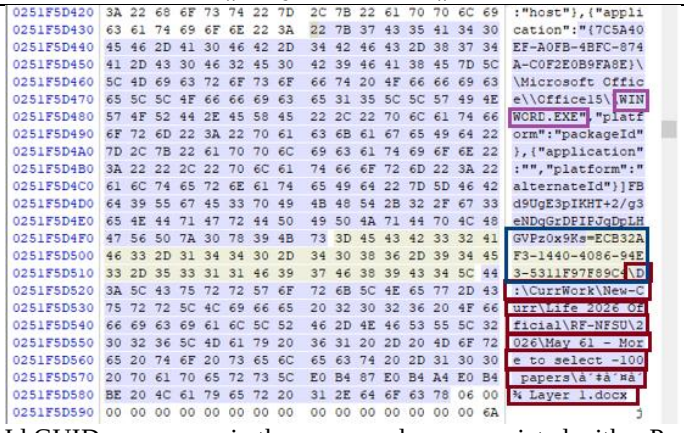

Figure 6: AppActivityId GUID occurrence in the memory dump associated with a PowerPoint presentation.

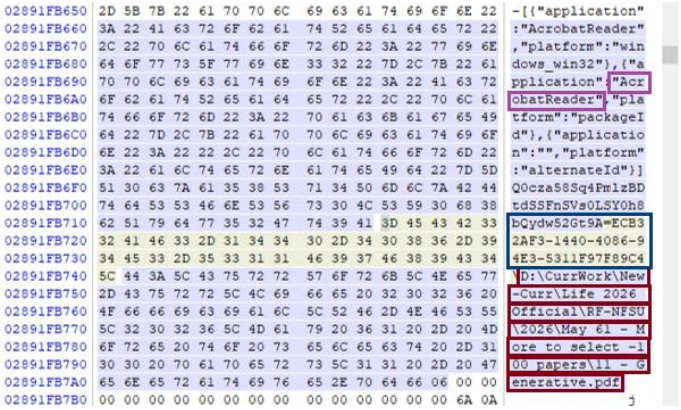

Figure 7: AppActivityId GUID occurrence in the memory dump associated with a PDF document (.pdf).

Another important finding of this research concerns the consistency of the AppActivityId value. Experiments were conducted on thirty-two different Windows systems, and the same AppActivityId value, **"ECB32AF3-1440-4086-94E3-5311F97F89C4"**, was observed in twenty-seven of these systems. This indicates that the identified GUID is not system-specific and appears to be consistently associated with Windows Activity History records across multiple Windows installations. The repeated occurrence of this value across a large number of systems suggests that it can serve as a reliable forensic artefact for automated memory analysis and file-system activity recovery. The recovered artefacts demonstrate the effectiveness of the proposed methodology in identifying recently accessed files directly from volatile memory. Even in situations where a file has been deleted, moved, or is no longer easily identifiable through conventional disk analysis, the presence of its filename and path information in memory can provide valuable evidence regarding its existence and usage on the system. Therefore, the methodology offers investigators a practical mechanism for rapidly narrowing the scope of forensic investigations by identifying a focused set of potentially relevant files from among the hundreds of thousands of files that may be present on modern storage devices. Overall, the experimental results confirm that Windows volatile memory contains valuable file-system activity artefacts that can be systematically recovered using the proposed AppActivityId-guided methodology. The recovered filenames, directory paths, and associated application information demonstrate the forensic value of these artefacts and their potential to complement traditional storage-based forensic investigations.

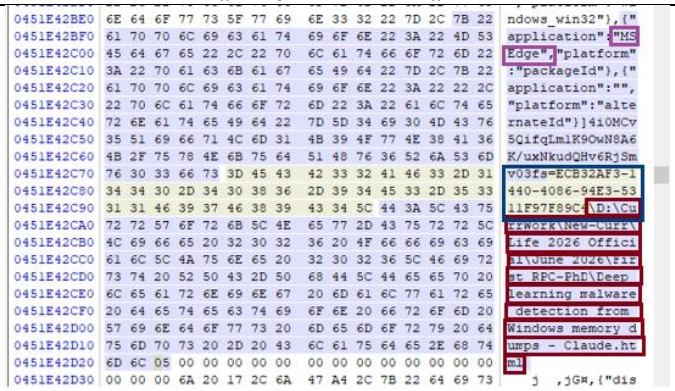

Figure 8: AppActivityId GUID occurrence in the memory dump associated with a multimedia video file (.mp4).

5. Challenges

Despite the significant forensic value of volatile memory analysis, the acquisition and examination of physical memory in Windows environments presents a range of substantial technical and procedural challenges that investigators must navigate to ensure the reliability and completeness of the evidence recovered.

A primary technical challenge arises from the inherent volatility and temporal inconsistency of physical memory. Since RAM is continuously modified by running processes, kernel operations, and hardware interrupts throughout the acquisition window, the resulting memory image does not represent a perfectly consistent system snapshot. Artifacts may be partially overwritten, memory pages may be in a transient state mid-operation, and the relative ordering of events captured within a single image may be difficult to establish with precision [13]. This temporal inconsistency is particularly significant when reconstructing file system activity sequences, where the accurate determination of access and modification order may be central to the investigative narrative.

The increasing adoption of memory protection mechanisms in successive Windows releases introduces further acquisition and analysis complexity [14].

6. Conclusion

This paper has presented a novel memory forensics methodology for the recovery of file system activity artifacts from Windows volatile memory. By targeting GUID-based searching unique to the Windows operating environment, the proposed methodology enables investigators to systematically recover the file name, the directory path of the accessed file, and the application used to access it — directly from a physical memory image. Experimental evaluation confirmed the effectiveness of the approach across representative investigative scenarios, demonstrating its capability to recover forensically significant file system activity records that would otherwise be inaccessible through traditional disk-based forensic methods.

The practical significance of the proposed methodology extends beyond the recovery of individual artifacts. Given that a Windows system disk may contain millions of files, the ability to extract a focused and actionable record of recently accessed or modified files from volatile memory substantially reduces the investigative scope, enabling examiners to direct analytical resources toward a forensically significant subset of the total data. This capability directly addresses one of the most persistent practical challenges in digital forensic investigations — the management of evidentiary scale — and represents a meaningful contribution to investigative efficiency and analytical precision.

The results of this research confirm that volatile memory analysis constitutes an indispensable and complementary evidence source in Windows digital investigations. In environments where persistent storage evidence is rendered inaccessible through full-disk encryption, selectively sanitized through anti-forensic tooling, or entirely absent due to fileless execution techniques, physical memory may represent the sole available repository of file system activity evidence. The integration of the proposed methodology with conventional offline forensic analysis therefore provides a more comprehensive and resilient investigative framework than either approach can offer independently.

References

[1] H. Nyholm, K. Monteith, S. Lyles, M. Gallegos, M. DeSantis, J. Donaldson, and C. Taylor, "The Evolution of Volatile Memory Forensics," Journal of Cybersecurity and Privacy, vol. 2, no. 3, pp. 556–572, 2022. doi: 10.3390/jcp2030028.

[2] M. Cohen, J. Hale, and M. Harlan, "Memory Analysis of .NET and .NET Core Applications," Forensic Science International: Digital Investigation, vol. 42, Art. no. 301404, 2022. doi: 10.1016/j.fsidi.2022.301404.

[3] I. Hamid and M. M. H. Rahman, "A Comprehensive Literature Review on Volatile Memory Forensics," Electronics, vol. 13, no. 15, Art. no. 3026, 2024. doi: 10.3390/electronics13153026.

[4] I. Kara, "Fileless Malware Threats: Recent Advances, Analysis Approach through Memory Forensics and Research Challenges," Expert Systems with Applications, vol. 214, Art. no. 119133, 2023. doi: 10.1016/j.eswa.2022.119133.

[5] A. A. Adebola, G. B. Akintola, and S. S. Shittu, "Performance Evaluation of Memory Forensic Tools for Extracting User Activity Artifacts from Windows 10 Memory Dump," International Journal of Scientific Research in Computer Science and Engineering, vol. 13, no. 6, 2025.

[6] M. Dener, G. Ok, and A. Orman, "Malware Detection Using Memory Analysis Data in Big Data Environment," Applied Sciences, vol. 12, no. 17, Art. no. 8604, 2022. doi: 10.3390/app12178604.

[7] A. Mishra and P. Bagade, "MalDicom: A Memory Forensic Framework for Detecting Malicious Payload in DICOM Files," arXiv Technical Report arXiv:2312.00483, 2023.

[8] S. Dija and A. R. Fernandez, "RAM Forensics of Various Versions of Windows OS Systems," International Research Journal of Engineering and Technology, vol. 11, no. 12, 2024.

[9] O. Khalid, I. Ullah, M. Ahmad, and R. Ahmad, "An Insight into the Machine-Learning-Based Fileless Malware Detection," Sensors, vol. 23, no. 2, Art. no. 612, 2023. doi: 10.3390/s23020612.

[10] S. L. Sanna, D. Maiorca, and G. Giacinto, "An Explainable Memory Forensics Approach for Malware Analysis," arXiv Technical Report arXiv:2602.19831, 2026.

[11] S. Schmitt, M. Stüttgen, and A. Dewald, "Windows Memory Forensics: Identification of (Malicious) Modifications in Memory-Mapped Image Files," Forensic Science International: Digital Investigation, vol. 45, Art. no. 301561, 2023. doi: 10.1016/j.fsidi.2023.301561.

[12] S. Dija, K. Lokeswari, and S. Shaji, "Uncovering Digital Evidence through Timeline Artifact Analysis in Windows OS Systems," International Journal of Recent Engineering Research and Development (IJRERD), vol. 10, no. 2, pp. 13–19, 2025. doi: 10.56581/IJRERD.10.02.13-19.

[13] A. Oliveri and D. Balzarotti, "A Comprehensive Quantification of Inconsistencies in Memory Dumps," arXiv Technical Report arXiv:2503.15065, 2025.

[14] T. Gharaibeh, I. Baggili, and A. Mahmoud, "On Enhancing Memory Forensics with FAME: Framework for Advanced Monitoring and Execution," Forensic Science International: Digital Investigation, vol. 49, Art. no. 101757, 2024. doi: 10.1016/j.fsidi.2024.101757.

Author Profile

Dija S received the B.Tech. Degree in Computer Science and Engineering in 1998 and the M.Tech. Degree in Software Systems in 2019. With over 25 years of experience in Digital Forensics Research and Development, she is currently serving as Scientist F at C-DAC Thiruvananthapuram under the Ministry of Electronics and Information Technology (MeitY), Government of India.