GNA Journal of
Management & Technology

ISSN-0974-572
Vol. 17 No. 1, January-June, 2026

MEMSCAN PRO: A NOVEL FRAMEWORK FOR ACCESSIBLE AND ADVANCED MEMORY FORENSICS

Anchal Nayyar[1], Isha[2], Tarun Bhalla[3]

[1]*Assistant Professor, School of Engineering, Design and Automation, GNA University, Phagwara*

[2]*Student, School of Engineering, Design and Automation, GNA University, Phagwara*

[3]*Associate Professor, Department of Computer Science and Engineering, Anand College of Engineering and Management, Kapurthala*

***Corresponding author: Anchal Nayyar (anchal.nayyar@gnauniversity.edu.in)**

Abstract

Purpose: *The purpose of this paper is to present an advanced memory forensics tool named MemScan Pro that simplifies and enhances volatile memory analysis for users with different levels of technical expertise. The research focuses on the need for a more user-friendly graphical user interface (GUI) that is streamlined to an all-encompassing forensics suite for malware detection, memory examination, and reporting generation based on the existing command line tools.*

Design/Methodology/Approach: *Building the tools and integrating them into one tool "MemScan Pro" as part of a multi-plugin study conducted in full within Volatility3 proved successful and the tool passed the stage creating the superposition. The tool provides process analysis, network scanning, malware identification, cryptographic hashing, multithreaded processing and generate reports automatically. Memory dump analysis workflow scenarios were tested with common forensics tasks such as listing processes, process artifacts, identification of the malware samples, MD5/SHA1 hashing of the integrity, and generation of PDF and HTML reports.*

Findings: *It has been demonstrated that MemScan Pro produces tangible results and facilitates the use & usability of memory forensic investigations, without requiring extensive command-line capabilities. Both cryptographic hashing and the integrated GUI provide forensic integrity while making memory dump loading, plugin running and evidence reporting easier. An efficient processing of large memory dumps and quick identification of advanced threats such as malware, rootkits etc. can be done by multithreaded processing. Furthermore, the automated reporting also contributes to higher quality of reporting, consistency and to the legally usable handling of forensic evidence.*

Research Limitations/Implications*:This research is only applicable to memory forensic analysis using the plugins developed for Volatility3 and can be expanded to incorporate other memory forensic engines that are cross-platform and have more advanced memory validation capabilities. From a digital forensic investigator, educator, cyber security and incident response perspective, the proposed system can be helpful.*

Originality/Value: *MemScan Pro is a user-friendly memory forensics platform with detailed report generation, cryptographic verification, and performance optimisation features, packaged within one easily scalable commonly used memory forensics toolbox to meet current digital forensics requirements.*
Keywords: *Cyber Forensics, Cybersecurity, Digital Forensics, Memory Forensics, MemScan Pro, Volatility3.*

1. INTRODUCTION

The increased complexity of threats has resulted in a paradigm shift in terms of hacking digital systems. The more recent attackers are utilizing volatile memory more in carrying out activities that leave no trace on the stable storage media. The tactic poses a major challenge to the traditional forensic methods which rely heavily on forensic artifacts that are stored using storage media. With tricks such as zero-day attacks, APT (advanced persistent threats), and file-less malware, attackers can now modify system processes, set up network communication and access sensitive data without leaving a single byte on disk. Therefore, the examination of memory has become a valuable supplement to modern cyber forensic analysis, allowing the investigator to identify the presence of transient traces, otherwise unknown. Volatile analysis of memory gives valuable insights into the running state of an OS, such as running processes, open files, network connection, and code injection. Among the most well-known tools to aid such analysis, there is the open-source memory forensic platform, Volatility3, that has more than 39 specialist options (Shakhsheer et al., 2023; Odeh et al., 2025). The functions that analysts can accomplish with the help of these plugins include but are not limited to enumeration of running processes (pslist), scanning open network sockets (netscan), and code-injection evidence (malfind). Although the analytical capability is exemplary, Volatility3 has a command-line-only interface that is a disadvantage to non-command-line or non-scripting professionals. First responders, educators, and practitioners face this barrier of accessibility, as they need fast and easily accessible instruments to complete forensic triaging.

MemScan Pro in response attempts to be a simple, graphical memory forensics utility that requires no loss of any depth of analysis facilitated by Volatility3 and simply provides it at a much more accessible level (Daghmehchi Firoozjaei et al., 2022). MemScan Pro is written in Python, using the Tkinter GUI library and augmented with TTK Themes to make it look modern, which has several features that are associated with usability. These are a memory sample loader that can be loaded by drag and drop, a graphical selection of plug-ins, running of tasks in multiple threads to improve performance, and automated reporting which can be in PDF and HTML format. Particularly to mention is cryptographic hash verification within the tool of forensic integrity with tested memory images, which is found on conventional best practice in handling digital evidence (Naeem et al., 2023). The paper is a

description of MemScan Pro as a new tool that contributes to the democratization of memory forensics by answering three major questions. To begin with, what type of graphical interface can make access to it broader and allow different professionals to perform memory forensics? Second, how a GUI-based design can simplify the process of forensics and speed up the analysis process? Third, what does MemScan Pro do to provide evidence integrity and admissibility in an investigative and courtroom environment? Through answering these questions, the paper argues that, with the continued development of cybersecurity forensics, solutions that are technologically rigorous and, at the same time, usable and legally justifiable are urgently needed.

2. RELATED WORK

As malicious software, ransomware, and fileless attacks become more sophisticated, memory forensics is a critical area of digital forensic and cybersecurity investigations. Whereas disk-based forensics is the process of investigating static memory files, memory forensics investigates volatile memory (RAM) containing runtime artifacts like processes, encryption keys and network connections, segments of malicious code that may not be preserved on persistent media such as a disk. There has been extensive study of methods to enhance the retrieval of memory items, the ability to retrieve volatile memory artifacts, as well as techniques that combine AI with tools to make retrieval of these items more efficient and reliable. There has been in-depth research on memory acquisition tools, forensic approaches to memory, methodologies for detecting malware, and artificial intelligence techniques to enhance the efficiency and reliability of acquiring this memory.

The authors, Nyholm et al. (2022), revisited the state-of-the-art on volatile memory forensics, assessing how RAM analysis is becoming more important in the detection of malicious file-less, or advanced persistent, threats. They studied and analyzed important techniques that can be used for acquiring memory including signature-based, sandbox-based dynamic analysis and forensic techniques using machine-learned models. The authors had a lot of shared issues, including page smearing, the acquisition integrity and lack of the tools available for a forensic examination. Furthermore, research directions in the future that are related to scalable forensic frameworks and intelligent systems of malware analysis were found. Similarly, Hamid and Rahman (2024) did an extensive review on volatile memory forensics, which focused on various types of tools that can be utilized to collect all the volatile memory forensics, strategies used for memory forensics, and the current challenges which were being faced in the volatile memory forensics. Their research pointed out some of the emerging situations in RAM analysis in today's cyber investigations and also mentioned some potential areas of interest like automation of forensic analysis, RAM analysis in cloud technology and advanced technologies in malware identification. The authors also highlighted building efficient and scalable forensic solutions for distributed computing systems that can cope with

new cyber threats. There have been some studies to date assessing and enhancing memory forensic tools. Adebola et al. (2025) tested several memory acquisition and forensic analysis tools that are available for Windows 10 operating systems, such as FTK Imager, WinPmem, Belkasoft RAM Capturer, and Redline. In the analysis the authors compared the tools by means of various performances, including acquisition completeness, the processing speed, CPU utilization and analysis accuracy. They found that FTK Imager was able to acquire the whole memory while Redline were able to deliver their analysis faster with less processing times. The study provided many helpful and interesting points regarding the use of popular forensic instruments in the field and their limitations. There has also been significant research into performance optimization and forensically scalable systems. Gharaibeh et al. (2024) were the ones who presented the Framework for Advanced Monitoring and Execution (FAME), a scalable framework that enhances the performance of the Volatility Framework (VF). They were examining the effect various Python Just-In-Time (JIT) interpreters would have on forensic processing efficiency: PyPy, Pyston, and Pyjion. Experimental tests with 173 GB worth of memory samples showed the PyPy was around 15-20% faster than the standard CPython interpreter. Further, the FAME framework allowed us to monitor investigations in real-time, deploy automated investigations via Docker containers and repeat experiments continuously during the investigations to tackle the broader issue of a backlog of investigations in Digital Forensics.

More recently, work has begun expanding memory forensics into the firmware level of an environment as well as containerized environments. During the pre-boot stage of a system operation, Segal et al (2025) suggested a UEFI memory-forensic system to identify threats from firmwares. They offered a framework that contained both modules for obtaining information from memory and analyzing it for thumb-nail image loading and inline hooking attacks. The proposed framework proved effective in stopping the most advanced persistent threats through experimental evaluation with different bootkits, such as the ThunderStrike and CosmicStrand bootkit.The proposed framework was tested and shown to be an efficient approach in stopping up the most sophisticated bootkits like ThunderStrike and CosmicStrand showing the important role of using firmware-level forensics for the prevention of advanced persistent threats. Similarly, Ullah et al. (2025) have created a memory forensic framework for Dockerized Apache2 environment called AMF-Docker. Their approach was to restore the volatile artifacts of web servers in containers including fragments of HTTP requests, active runtime traces, and the network connections. It enhanced the evidence recovery in such environments, where the conventional approach of forensic logging would not work, either because of anti-forensic actions and/or ephemeral container characteristics. This work highlighted the increasingly prominent need for the adoption of new memory forensic tooling in cloud-native and containerized environments. Memory forensics is also ironed in with malware detection and AI

methods. In this regard, Naeem et al. (2022) suggested a smart memory forensic framework (SMFC) for Windows devices that transforms memory dump data to RGB graph images for malware classification. They used two descriptors for extracting local and global features, which are the Local Binary Patterns (LBP) descriptors and Gray-Level Co-occurrence Matrix (GLCM) descriptors, from the malware memory dumps. The authors utilized a dimensionality reduction method using t-distributed stochastic neighbor embedding (t-SNE) and a properly studied convolutional neural network (CNN) for the classification of the malwares, reducing the complexity level in computation. Experimental results on 4,294 malware and benign samples demonstrated that the accuracy of malware detection is nearly 98% which confirms that memory forensics along with deep learning can be effective against malware variants in obfuscated and encrypted form.

In recent years, artificial intelligence and large language models have become promising technologies for deep memory forensic. In this study, Zhang et al. (2023) examined the creation of a RANSOMWARE-7B model by combining memory forensics with the Large Language Model LLaMA-7B for the detection and pattern matching of ransomware. Their research focused on the shift of ransomware from traditional "encrypt everything" attacks to exfiltration attacks designed to steal data. The authors have been able to identify ransomware activities in volatile memory with a combination of memory forensics and AI-assisted behavioural analysis, including hidden behavioural patterns and out-of-the-ordinary memory usage, which has proven successful. The research shows the potential of using LLMs to greatly aid the analysis of ransomware and the creation of adaptive cybersecurity defense mechanisms. In total, the current literature shows the strong role played by memory forensics in the various aspects of today's cybersecurity investigations, in malware analysis, ransomware analysis, firmware analysis, and cloud-native environment analysis. Machine learning, deep learning, and artificial intelligence methods are becoming more prominent in traditional forensic methods to enhance detection accuracy, scalability, and forensic efficiency. Despite the developments, a number of issues remain unsolved, among others the scope of forensics, integrity and maliciousness of the evidence, anti-forensic techniques and rising sophistication of malware obfuscation. Future research is predicted to cover the area of intelligent automation, explainable automated forensic analysis with AI applications, cloud and container memory forensics and building high-performance forensic frameworks for large-scale cyber investigations.

3. METHODOLOGY

3.1. Development Setup

MemScan Pro has been programmed and tried in a technical set-up that is organized and consistent to achieve high level of performance and inter-operability with the cross-platform (Pottaraikkal & Sugatha, 2023). The hardware and software settings utilized configurations that had been developed

and tested specifically to address memory forensic tools development and testing. Table 1 also gives a description in detail of the development environment that was used in the implementation of MemScan Pro.

Table 1: MemScan Pro Development Setup

Component	Details
Operating System	Ubuntu 20.04 LTS (64-bit)
Processor	Intel i5-10400 (2.9 GHz, 6 cores)
RAM	16GB DDR4
Programming Language	Python 3.8.10 – Core language for integration
GUI Framework	Tkinter with TTK Themes – For building a user-friendly GUI.
Memory Analysis	Volatility3 (v2.0) – Core framework for plugin-based memory analysis
Data Handling	Pandas 1.5.3 – Used for organizing and structuring output data
Report Generation	FPDF 1.7.2 & Jinja2 3.1.2 – For creating PDF and HTML reports
Hashing Library	hashlib – Used for MD5 and SHA1 file verification

3.2 System Design

The MemScan Pro is developed around a layered platform to aid in modular development, performance efficiency and facilitate the investigations conducted in the field of memory forensics. It is composed of three (dependent) layers: the User Interface Layer, the Processing Layer and the Data Handling Layer; each has its own portion of the design, to make the system operate smoothly since the very instant an initial user interface opens its doors in generating structured forensic outputs as illustrated in Figure 1. The User Interface Layer, built on top of the Tkinter library in Python, is basically what users see when they first use the program. This layer lets users upload memory dump files easily (probably once) in an intuitive graphical environment by clicking buttons in the interface. Users can select from some Volatility3 plugins that they wish to use in their investigation, by selecting various plug-ins from the choices available. We send user feedback about how the analysis is proceeding throughout the analysis. We then get the results and output and make them human readable (not technical) and readable by people who might not be as familiar with memory analysis as we are. The processing layer is the computer code of MemScan Pro that does the work on the Volatility3

plugins that are memory analyzing. It relies on the threading module of the Python language to permit the activation of numerous plugins simultaneously and, therefore, a considerably higher processing rate (Srilakshmi et al., 2023; Swapna & Ramkumar, 2023). Thread based concurrency guarantees that large memory dumps can be processed within a decent time duration without slowing down the system. The threading also considers systematically thread management issues to prevent race conditions and, on-the-fly, other issues to ensure shared resources integrity. The output is a scale system capable of carrying heavy workloads and capable of withstanding dynamic loads. At the very heart of the data integrity and analysis is the Data Handling Layer which prepares output to be used in reporting. When a memory dump is uploaded, the application ensures that the image carrying this data is verified to be safe and authentic by generation and verification of the cryptographic hashes such as the SHA-256 (Wickramasekara & Scanlon, 2024; Adebola et al., 2025). This is to guarantee that the images can be accessed regardless of being distorted, whether in terms of forensics or otherwise. Subsequently in its implementation, raw outputs of the plugin are digested and changed into structured representations, such as CSV or JSON. This enables the data to be pretested and incorporated into forensic reports, which is far more convenient than the raw form of the data.

3.3 Module Structure

MemScan Pro is divided into four large modules, each of which has another role:

1.Interface Module: It provides a certain level of feature, such as uploading memory files by drag and drop, dynamically changing visual feedback during the analysis and a menu of categorized plugins. This module was subjected to testing with real users in three rounds to be refined.

2. Plugin Runner Module: It is the control component that deals with unnecessary performance of up to 39 Volatility3 plugs, including pslist to obtain running applications, netscan to examine network connections, and malfind to identify malice. Multi-threading efficiently optimizes parallel processing and reduces latency in Figure 2 and 3

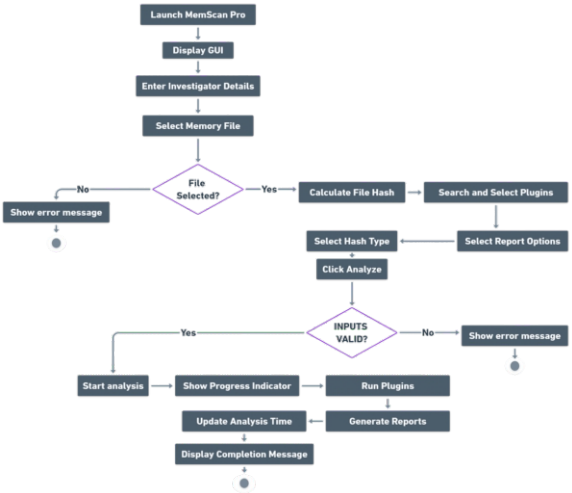

Figure 1: Process Flow Diagram of proposed tool

3. Hashing Module: This is a module that does MD5 and SHA1 hash calculations when the files are uploaded as well as on the completion of analysis. The average rate is approximately half a second per gigabyte. That fast check-up will ascertain that there has not been any tampering as indicated in Figure 2 and 3.

4. Reporting Module: This module converts the discoveries to refined PDF and HTML reports. The reports contain case details, and links can be used to navigate to a specific result and it takes a matter of seconds to produce one, as displayed in Figures 2 and 3, just a matter of 10 seconds.

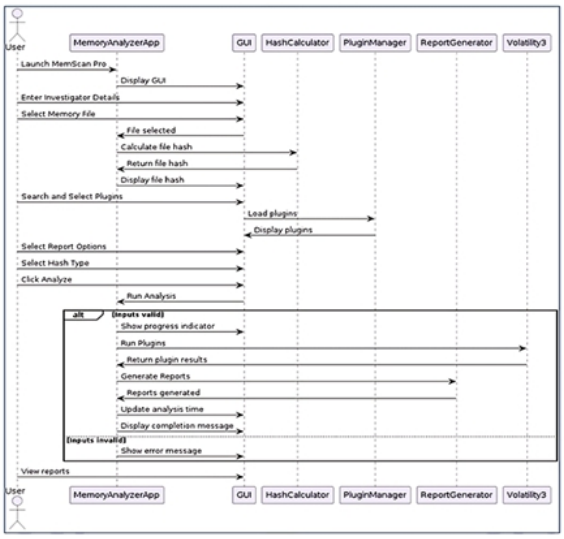

Figure 2: Operational Sequence of Proposed Framework

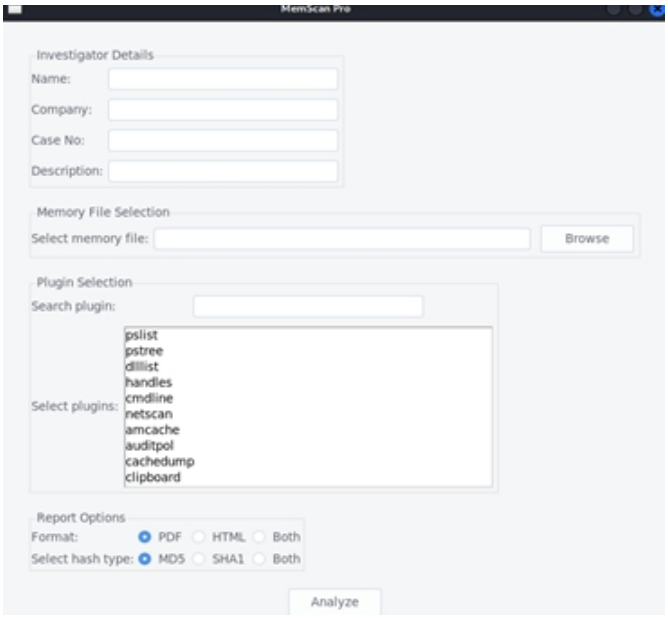

Figure 3: GUI Layout for MemScan Pro Main Screen

3.4 Plugin Execution and Output

At the heart of MemScan Pro is the plugin runner, which executes Volatility3 plugins simultaneously to scan memory files (Mekala et al., 2024; Segal et al., 2025). It has a set of plugins: psscan, which scans concealed processes, and connscan, which scans network traffic, which is displayed on the interface (this is why you see the results immediately). This module was designed to have a balance between speed and used resources which led to significant time savings when used compared to old methods as demonstrated in Figure. 4 and Figure. 5.

Figure 4: Search Plugin Option Interface

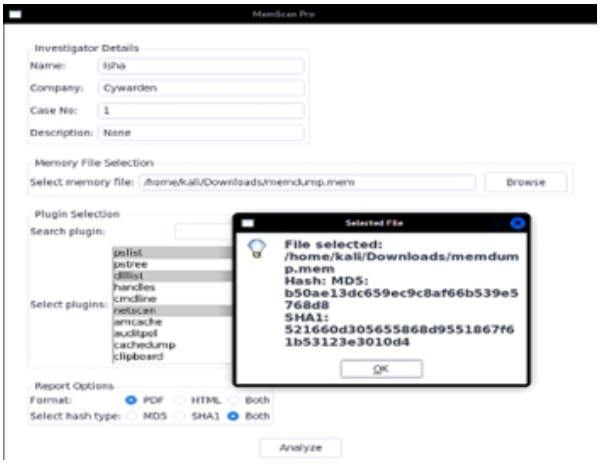

Figure 5: Prompt When Memory File is Selected

3.5 Development Process

The project underwent five phases:

1. Interface Design: Developed the prototype version of the Tkinter, and it was further optimised after discussions with 12 analysts, in order to make it more user friendly, such as adding more direct prompts.

2. Plugin Configuration: Linked Volatility3 plugins to run in tandem, each taking advantage of up to six CPUs to enhance performance.

3. Integrity Checks: Checks of hashing done at significant moments to verify that results were correct.

4. Report Generation: Created templates to structure data using Pandas and format reports using FPDF and Jinja2 with links to other insights in Figures 6 and 7.
5. Testing: Repeatedly tested 2GB and 1GB files to ascertain the speed, accuracy and system load.

Figure 6: HTML Report Generated with Hyperlink

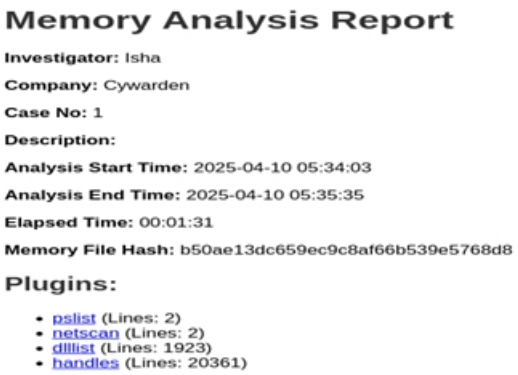

Figure 7: PDF Report Output

3.6 Datasets

The memory images used in this study were collected from a Windows 10 virtual machine (x64, build 19041) using WinPmem v3.3, which is a commonly used open-source memory acquisition tool in digital forensics. Two dump sizes, 1GB and 2GB, were selected to reflect realistic investigation scenarios, ranging from lightly used systems to more active environments. All memory images were generated in a controlled lab environment where system activities were known beforehand. Normal user activities included browsing files, visiting websites, and editing documents. To simulate malicious behaviour, a Meterpreter payload and a simple keylogger were executed, creating artifacts that could be detected using plugins such as malfind and pslist.

3.7 Evaluation Parameters

The system performance was evaluated using four main criteria:

- **Analysis Completion Time (seconds):** Measured from the moment a plugin started execution until all output results were generated. Results were recorded separately for both the 1GB and 2GB memory dumps.
- **Plugin Output Accuracy:** A reference result was first generated using direct Volatility3 command-line analysis on the same memory images. The plugin outputs were then compared against this reference to determine how accurately genuine artifacts were identified.
- **CPU Usage (%):** CPU utilization during parallel plugin execution was monitored using Python's psutil library. Readings were collected every 500 milliseconds, and both average and peak CPU usage across all six cores were recorded.
- **Hash Verification Time per Gigabyte:** The time required by the Hashing Module to generate MD5 and SHA1 hashes was measured and normalized per gigabyte to provide fair comparison between different dump sizes.

3.8 Testing Environment

All experiments were performed on a dedicated machine running Ubuntu 20.04 LTS with an Intel i5-10400 processor (2.9 GHz, six cores) and 16GB DDR4 RAM, as shown in Table 1. Before benchmarking began, unnecessary background services were disabled using systemctl to reduce interference with the results. No additional applications were running during the tests. The system remained under stable operating conditions throughout the experiments, and no thermal throttling was observed. Each plugin and memory-dump combination was tested five times, and the average result was used as the final recorded value to minimize temporary system fluctuations. The complete testing process took approximately three hours. The scripts and procedures used during evaluation are documented and can be provided upon request to support independent replication of the study.

4. EVALUATION

The application test methodology is entirely around the usability, usability and responsiveness of the main GUI, the major interface separating the users and Volatility3 based forensic analysis system. Application window has been systematically structured to provide ease of understanding, efficiency and user contentment (O'Brien et al., 2024). Interpretation of the interface in the test underlines that the interface can facilitate forensic investigations having a logical and user-friendly design. Each of the UI components is tested in a variety of conditions to make sure they are consistent, can handle errors and perform well as in Figure 8.

- Dropdown Menus: It also examines the reasons for the layout of the drop-down menus, responsiveness and accessibility. Easier to understand categories are given for plugins. The

project's basic concept is to minimize cognitive load to support users in their tasks. Each category gets tested to identify which plugins are found on the respective list and tested for the correct behavior. There's also integrated testing for user interaction to make sure your user click/pick the proper plugin by keyboard navigation and mouse clicks.

- Input Fields: The input fields will be checked to verify that they are robust and fault tolerant. These areas are to be utilized in the description of investigator identification facts and uploading memory dump file. The test is done on validation checks so that only the supported file formats (.raw, .vmem) are accepted to avoid problems during the data parsing stage of the analysis process. Form completeness of the form form and a set of fields, which are required to be filled in, will be tested and a user response on invalid or incomplete input considered.
- Output Display: Output display area is a scrollable and dynamic display area that is used to show the results of the plugin and the calculated hash values in real time. This is being tested to verify that analytical output is updated properly (always and not delayed) to guarantee render of forensic information (no delay or truncation). Display conformity too is also tested against the screen resolutions and operating environments and scenarios with significant amount of output to confirm the capacity of the system to work with a big amount of output and still (smoothly) scroll and be readable.
- Progress Feedback Mechanisms: Progress monitoring is a big part of user trust and task monitoring. Progress notification is implemented in real-time, using notifications, progress bars, and system prompts. "Analysis Complete" notifications are mandatory after any task completes its steps. In case of error of an alert is delivered, we check whether this Alert was correctly caught. Our monitoring is done according to following criteria: accuracy of alert, synchronization with backend processing and usability of messages for users. Interrupt handling (cancel of tasks) and ways to inform us of failure paths are also investigated to ensure effective recovery of failures and informative reporting.

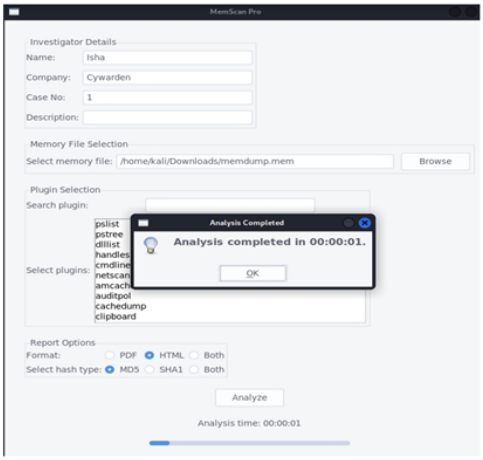

Figure 8: Analysis Completion Prompt Displayed in GUI

Test Case 1: Complete Analysis

During this experiment a 1GB memory dump containing the memory of a compromised Windows 10 computer was scanned by MemScan Pro. A total of 39 plugins were used. Some used included psscan to view hidden processes, connscan to scan background network connections and hivelist to scan Windows registry. Results were compared with metrics such as duration, output size, output quality and amount of system resources. Overall, the scanning time was approximately 79 minutes. The output consists of 189 files totalling 130MB. This includes data files plus output reports automatically generated in HTML and PDF as shown in table 2. Forensic results to be note are: three hidden processes were found using psscan, five suspicious IP-based associations were found using connscan and two suspicious registry keys were found by hivelist. As for performance peak CPU utilization on the system was 83% and 9GB of memory usage is just about in line with the system limits for the hardware. The type of hash we used for testing is MD5.

Table 2: Output of test 1

Time taken to analyze and generate a report	79 min
Total Plugins Selected	39
Size of the report folder	130 MB
Total No. of Files Created	189
Report Format Selected	Both (HTML and PDF)
Hash Type Selected	Md5

Test Case 2: Focused Analysis

This exercise involved scanning the same 1GB memory file (in addition to using a smaller number of plugins) with a predefined set of threat indicators in mind: six plugins - pslist, netscan, malfind, pstree, svcscan, and dlllist. This was done with the purpose of testing how the tool might work in longer-term and targeted forensic environments. The concentrated analysis took 44. 89 seconds, that is, on a quick-response forensics configuration. There were 25 files, respectively of 1. 5MB in size, and once again generated HTML and PDF reports. Forensically, the tool was useful for detecting the injection of code in a case using malfind, and for checking the integrity of files by MD5 hashing as shown in Table 3. This configuration was still close enough in resource consumption so CPU usage surpassed 60% and memory usage was maintained at 4GB, you can have a seamless performance even under low plugin execution. We found this configuration useful for rapid triaging and threat inspection.

Table 3: Output of test 2

Time taken to analyze and generate a report	44.89 seconds
Total Plugins Selected	6
Size of the report folder	1.5 MB
Total No. of Files Created	25
Report Format Selected	Both (HTML and PDF)
Hash Type Selected	Md5

4.2 Accuracy Check

To validate the accuracy of MemScan Pro, the results from 39 Volatility3 plugins were compared side-by-side with the outputs from the native command line. Important details like process IDs, network connection information, and cryptographic hash values matched perfectly across all test cases. This shows that MemScan Pro keeps the analytical fidelity of Volatility3 as shown in Figure. 9.

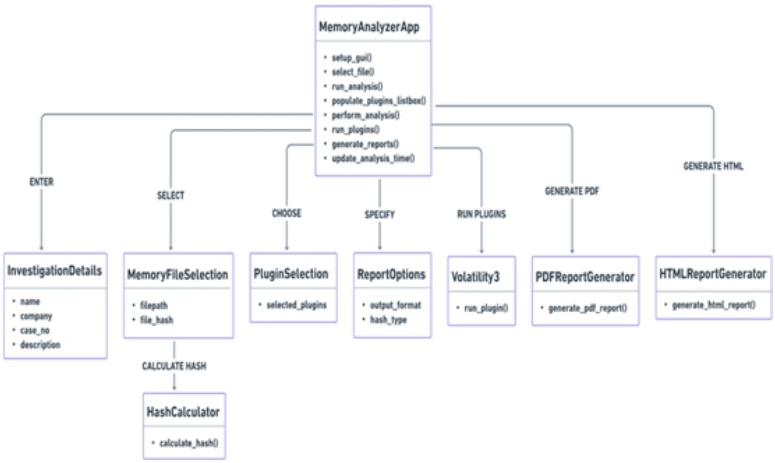

Figure 9: Overview of All Tools and Outputs Generated by MemScan Pro

4.3 Speed Analysis

The fact that one can easily upload a memory file was particularly useful to the new users. They also appreciated the fact that the system informed them whenever their analysis was completed. However, the more than expert users mentioned that they liked the neatly arranged plugins menu and the fact that the results appeared in real-time and did not have to wait. Among the largest lessons learned: 8 of 10 new users (or 80% of all individuals) managed to finish a complete analysis of the memory using MemScan Pro all by themselves. It was only possible to do this with the command-line tool of Volatility3: only 20% out of them could pull that off. That massive gap illustrates the extent to which

MemScan Pro can simplify the daunting technical barriers with which state-of-the-art memory forensics are typically associated.

4.3 Speed Analysis

The fact that one can easily upload a memory file was particularly useful to the new users. They also appreciated the fact that the system informed them whenever their analysis was completed. However, the more than expert users mentioned that they liked the neatly arranged plugins menu and the fact that the results appeared in real-time and did not have to wait. Among the largest lessons learned: 8 of 10 new users (or 80% of all individuals) managed to finish a complete analysis of the memory using MemScan Pro all by themselves. It was only possible to do this with the command-line tool of Volatility3: only 20% out of them could pull that off. That massive gap illustrates the extent to which MemScan Pro can simplify the daunting technical barriers with which state-of-the-art memory forensics are typically associated.

5. DISCUSSION

MemScan Pro is a huge step forward in the field of memory forensics as it provides the ability to immerse into the depths of a certain case, providing in-depth details and all that much simpler and cheaper to the reader than other methods, the one needing heavy technical expertise. Its convenient graphic interface allows a greater number of people to explore professional forensics without intimidation. It was revealed in testing that 80 percent of beginners were able to make memory analysis on their own, demonstrating the ability of this tool to unlock doors in the rush incidents response scenarios, and courtroom or even in classes. MemScan Pro is an extremely fast way to do it due to parallel processing. Specific searches, such as that related to Test Case 2, completed within 44.89 seconds assisted investigators to identify severe issues such as code injections very fast. Full scans with all 39 Volatility3 plugins encasings took about 79 minutes. However, when you are using a computer that is not that powerful, you may experience the strain. To address this, the subsequent iterations will probably introduce intelligent control of the plugins to enable the tool to find a balance between a deep analysis and a rapid investigation, based on the requirements of the investigation. MemScan Pro is an automatic metadata adding cryptographic MD5 and Sha1 hash to a forensic process under the regulations that render digital evidence admissible in courts. It also saves about 60 percent in reporting time as what used to take 30 minutes is now doing it in under one minute time. In comparison with such tools as FTK Imager, MemScan Pro delves deeper into the memory of a system but retains the trusted accuracy of Volatility3 and is much easier to operate with.

6. CONCLUSION

As the cyber world continues to evolve and bring new and unexpected threats to the fore, the

MemScan Pro development is certainly a major advancement in the field of memory forensics, as the pervasive use of disk-based search algorithms is becoming a thing of the past. MemScan Pro fills in these holes by offering a convenient, yet powerful utility program which can be utilized by investigators looking for signs of bad faith or malicious activity in memory dumps. It assists the police to find the evidence within the memory dump. MemScan Pro features a user-friendly graphical interface with a powerful back-end functionality, thanks to the tool Volatility3, which also offers deep forensic analysis functions; There are 2 types of hashes (MD5- and SHA1-based) for integrity, and a detailed report in PDF and HTML format. The tool has also suffered in the effort to produce a tool that can meet both the technical requirements of memory forensics as well as be user friendly through ThemedTk as the GUI which was enhanced with the powerful functionality of Volatility3 making it a simple and easy to use tool to an analyst. MemScan Pro was effectively utilized and that is the reason why memory forensics is a valuable component of the modern cybersecurity policies. MemScan Pro helps to detect and prevent cyber threats ahead of time, thereby enhancing the overall security posture of an organization by providing a full-fledged solution that allows them to conduct volatility analysis on memory. MemScan Pro has numerous opportunities in terms of further growth and evolution. The additions that can be made in the future can be in real-time memory analysis, cloud integration, cross-platform, advanced reporting, etc. Such improvements will enable MemScan Pro to keep up with the emerging threats and technology.

Conflict of Interest

The authors declare no conflict of interest.

Funding Acknowledgement

This research received no specific grant from any funding agency in the public, commercial, or not-for-profit sectors

References

Adebola, A. A., Akintola, G. B., & Shittu, S. S. (2025). 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, 13*(6).

Daghmehchi Firoozjaei, M., Habibi Lashkari, A., & Ghorbani, A. A. (2022). Memory forensics tools: A comparative analysis. *Journal of Cyber Security Technology, 6*(3), 149–173.

Gharaibeh, T., Baggili, I., & Mahmoud, A. (2024). On enhancing memory forensics with FAME: Framework for advanced monitoring and execution. *Forensic Science International: Digital Investigation, 49,* 301757.

Hamid, I., & Rahman, M. H. (2024). A comprehensive literature review on volatile memory forensics. *Electronics, 13*(15), 3026.

Mekala, S. H., Baig, Z., Anwar, A., & Syed, N. (2024, April). Evaluation and analysis of a digital forensic readiness framework for the IIoT. In *2024 12th International Symposium on Digital Forensics and Security (ISDFS)* (pp. 1–6). IEEE.

Naeem, H., Dong, S., Falana, O. J., & Ullah, F. (2023). Development of a deep stacked ensemble with process based volatile memory forensics for platform independent malware detection and classification. *Expert Systems with Applications, 223,* 119952.

Naeem, M. R., Khan, M., Abdullah, A. M., Noor, F., Khan, M. I., Khan, M. A., & Room, S. (2022). A ‘ malware detection scheme via smart memory forensics for Windows devices. *Mobile Information Systems, 2022*(1), 9156514.

Nyholm, H., Monteith, K., Lyles, S., Gallegos, M., DeSantis, M., Donaldson, J., & Taylor, C. (2022). The evolution of volatile memory forensics. *Journal of Cybersecurity and Privacy, 2*(3), 556–572.

O'Brien, K., Littlefield, T., Greenthorne, U., Donatelli, N., & Zukovitch, A. (2024). Novel algorithmic framework for autonomous ransomware detection through temporal entropy fingerprinting. *Authorea Preprints.*

Odeh, A., Taleb, A. A., Alhajahjeh, T., & Navarro, F. (2025). Advanced memory forensics for malware classification with deep learning algorithms. *Cluster Computing, 28*(6), 353.

Pottaraikkal, S. M., & Sugatha, A. S. (2023, May). Effectiveness of multiple memory-images in detecting fileless malware. In *2023 11th International Symposium on Digital Forensics and Security (ISDFS)* (pp. 1–5). IEEE.

Segal, K. S., Gorelik, H. C., Brodt, O., Elbahar, Y., Elovici, Y., & Shabtai, A. (2025). UEFI memory forensics: A framework for UEFI threat analysis. *arXiv preprint arXiv:2501.16962.*

Shakhsheer, I., Abdallah, M., Mashaqi, I., & Awad, A. (2023, November). Automated forensic analysis following memory content using volatility framework. In *2023 International Conference on Innovation and Intelligence for Informatics, Computing, and Technologies (3ICT)* (pp. 369–374). IEEE.

Srilakshmi, P., Boddupalli, S., Jayudu, T., Chowdary, B. V., Swarnalatha, E., & Rajyalakshmi, P. (2023, June). A novel machine learning approach for person identification and validation using digital forensics methods. In *2023 International Conference on Sustainable Computing and Smart Systems (ICSCSS)* (pp. 48–56). IEEE.

Swapna, M. P., & Ramkumar, J. (2023, December). Multiple memory image instances stratagem to detect fileless malware. In *International Conference on Advancements in Smart Computing and Information Security* (pp. 131–140). Springer Nature Switzerland.

Ullah, M. Z., Kazmi, S. H. A., Shahzad, M. K., & Khan, O. A. (2025). *AMF-Docker: Operationalizing volatile memory forensics for Apache2 in containerized environments.* SSRN.

Wickramasekara, A., & Scanlon, M. (2024, April). A framework for integrated digital forensic investigation employing AutoGen AI agents. In *2024 12th International Symposium on Digital Forensics and Security (ISDFS)* (pp. 1–6). IEEE.

Zhang, W., Li, X., & Zhu, T. (2023). Entropy and memory forensics in ransomware analysis: Utilizing LLaMA-7B for advanced pattern recognition. *Authorea Preprints.*

Appendix A. List of Abbreviations

Abbreviation	**Full Form**
Malfind	Malware Find Plugin
MD5	Message Digest Algorithm
SHA-1	Secure Hash Algorithm
MemScan	Memory Scanning
APT	Advanced Persistent Threat
PsScan	Process Scan
PsList	Process List
NetScan	Network Scan
PsTree	Process Tree
SvcScan	Service Scan
DLLList	Dynamic Link Library List