International Journal of Computer Science & Communications
Vol. 11, No. 1, 2026, pp. 30-41
http://ijcsc.ielas.org/index.php/ijcsc
ISSN: 2545-4153

AUTOMATED RAM MEMORY FORENSIC TOOL FOR MALWARE ANALYSIS AND THREAT DETECTION

Tejavath Yogeswar Naik[1*], KP Supreethi[2]
Department of Computer Science and Engineering
University College of Engineering Science and Technology, Jawaharla Nehru Technological University, Kukatpally, Hyderabad, Telangana, India
*Corresponding Author: t.yogeshwarnaik@gmail.com[1], supreethi.pujari@jntuh.ac.in[2]

Received: 2026-08-21 Accepted: 2026-09-12 Published online: 2026-09-27

Abstract

Advanced cyber threats such as fileless malware, ransomware, and process injection attacks often reside in volatile memory, making them difficult to detect using traditional disk-based forensic techniques. This paper presents an Automated RAM Memory Forensic Tool for Malware Analysis and Threat Detection that integrates memory acquisition, forensic artifact extraction, signature-based detection, and artificial intelligence into a unified framework. The proposed system utilizes WinPmem for memory acquisition, Volatility 3 for extracting forensic artifacts, and YARA rules for identifying known malware signatures. Machine Learning and Deep Learning models are employed to detect unknown and obfuscated malware, while an ensemble decision mechanism improves detection accuracy and reduces false positives. The framework also performs automated IOC correlation, severity assessment, timeline reconstruction, and forensic report generation through a Streamlit-based interface. Experimental results demonstrate that the proposed approach improves malware detection efficiency, reduces manual investigation effort, and provides an effective solution for modern memory forensic analysis and incident response.

Keywords: Memory Forensics, RAM Analysis, Malware Detection, Volatility 3, WinPMEM, YARA, Machine Learning, Deep Learning, Digital Forensics, Threat Detection.

1. INTRODUCTION

The rapid growth of cyber threats has significantly increased the complexity of digital forensic investigations. Modern malware, including fileless malware, ransomware, rootkits, and Advanced Persistent Threats (APTs), increasingly operates within volatile memory, leaving minimal or no traces on persistent storage. As a result, conventional disk-based forensic techniques are often insufficient for detecting sophisticated attacks that execute entirely in Random Access Memory (RAM). Memory forensics has therefore emerged as a critical area of cybersecurity, enabling investigators to analyze volatile memory and recover valuable forensic artifacts such as running processes, loaded modules, network connections, registry information, encryption keys, and injected code.

Although memory forensic tools such as WinPmem and Volatility 3 provide powerful capabilities for memory acquisition and artifact extraction, the investigation process remains largely manual and time-consuming. Security analysts must examine thousands of extracted artifacts, correlate Indicators of Compromise (IOCs), identify malicious behavior, and generate forensic reports. This manual approach increases investigation time, requires extensive expertise, and may result in missed evidence or false-positive detections, particularly when analyzing large-scale memory images. Recent advances in Artificial Intelligence (AI), Machine Learning (ML), and Deep Learning (DL) have demonstrated promising capabilities for automated malware detection. ML and DL models can identify hidden behavioral patterns within forensic artifacts and detect both known and previously unseen malware that may evade traditional signature-based techniques. However, many existing solutions focus only on individual detection methods and do not provide a unified framework that integrates memory acquisition, forensic analysis, intelligent threat detection, and automated reporting. To address these challenges, this paper presents an Automated RAM Memory Forensic Tool for Malware Analysis and Threat Detection. The proposed framework integrates WinPmem for volatile memory acquisition, Volatility 3 for forensic artifact extraction, YARA rules for signature-based detection, and Machine Learning and Deep Learning models for intelligent malware classification. An ensemble decision mechanism combines multiple detection techniques to improve detection accuracy while reducing false-positive rates. The framework further performs automated IOC correlation, severity assessment, timeline reconstruction, and forensic report generation

2. RELATED WORK

Memory forensics has become a vital area of digital forensics due to the rapid evolution of sophisticated cyber threats that exploit volatile memory to conceal malicious activities. Modern attacks such as fileless malware, ransomware, rootkits, and Advanced Persistent Threats (APTs) often execute entirely in Random Access Memory (RAM), leaving minimal evidence on persistent storage. Consequently, traditional disk-based forensic techniques are no longer sufficient to detect these threats, making RAM analysis an essential component of incident response and malware investigation.

Several memory forensic frameworks have been developed to extract and analyze volatile memory artifacts. Tools such as Volatility, Volatility 3, Rekall, and WinPmem have become widely adopted in forensic investigations because of their ability to recover critical information, including running processes, loaded Dynamic Link Libraries (DLLs), registry hives, network connections, process handles, kernel objects, and injected code. These tools provide investigators with valuable evidence for reconstructing system activities and identifying malicious behavior. However, they primarily function as forensic analysis tools and require extensive manual interpretation of extracted artifacts.

As the size and complexity of memory images continue to increase, manual investigation becomes time-consuming, error-prone, and highly dependent on the expertise of forensic analysts.

To overcome these challenges, recent research has focused on integrating Artificial Intelligence (AI) into memory forensic analysis. Machine Learning (ML) algorithms such as Random Forest, Support Vector Machine (SVM), Decision Tree, and XGBoost have been employed to classify malicious processes using features extracted from memory artifacts. These techniques have demonstrated improved detection performance by identifying suspicious behavioral patterns rather than relying solely on predefined malware signatures. Furthermore, Deep Learning (DL) models, including Convolutional Neural Networks (CNNs) and other neural network architectures, have shown promising results in automatically learning complex relationships within memory data, enabling the detection of previously unseen and obfuscated malware variants.

In addition to AI-based approaches, signature-based detection techniques remain an important component of malware analysis. YARA rules are widely used to identify known malware families by matching characteristic patterns and signatures within memory artifacts. Although YARA provides accurate detection for known threats, its effectiveness is limited against zero-day attacks, polymorphic malware, and heavily obfuscated malicious code. Consequently, researchers have proposed combining signature-based techniques with AI-driven detection models to improve both detection accuracy and adaptability to evolving cyber threats. Recent studies have also explored explainable AI, graph-based learning, and automated forensic visualization to enhance malware analysis andimprove the interpretability of detection results. These approaches assist investigators in understanding the reasoning behind detection decisions and support faster incident response. However, many existing solutions focus on only one stage of the forensic workflow, such as memory acquisition, artifact extraction, malware classification, or reporting. Few systems provide a comprehensive framework that integrates all these components into a single automated platform.

Based on the identified research gaps, this work proposes an Automated RAM Memory Forensic Tool for Malware Analysis and Threat Detection that combines WinPmem for memory acquisition, Volatility 3 for forensic artifact extraction, YARA for signature-based malware detection, and Machine Learning and Deep Learning models for intelligent threat classification. An ensemble decision mechanism integrates the outputs of multiple detection techniques to improve classification accuracy and reduce false-positive rates. Furthermore, the framework performs automated Indicator of Compromise (IOC) correlation, timeline reconstruction, severity assessment, and forensic report generation through an interactive Streamlit dashboard. By integrating the complete memory forensic workflow into a unified framework, the proposed system reduces manual investigation

International Journal of Computer Science & Communications
Vol. 11, No. 1, 2026, pp. 30-41
http://ijcsc.ielas.org/index.php/ijcsc
ISSN: 2545-4153

effort, improves detection efficiency, and provides comprehensive support for modern digital forensic investigations and cyber threat response.

3. SYSTEM ARCHITECTURE

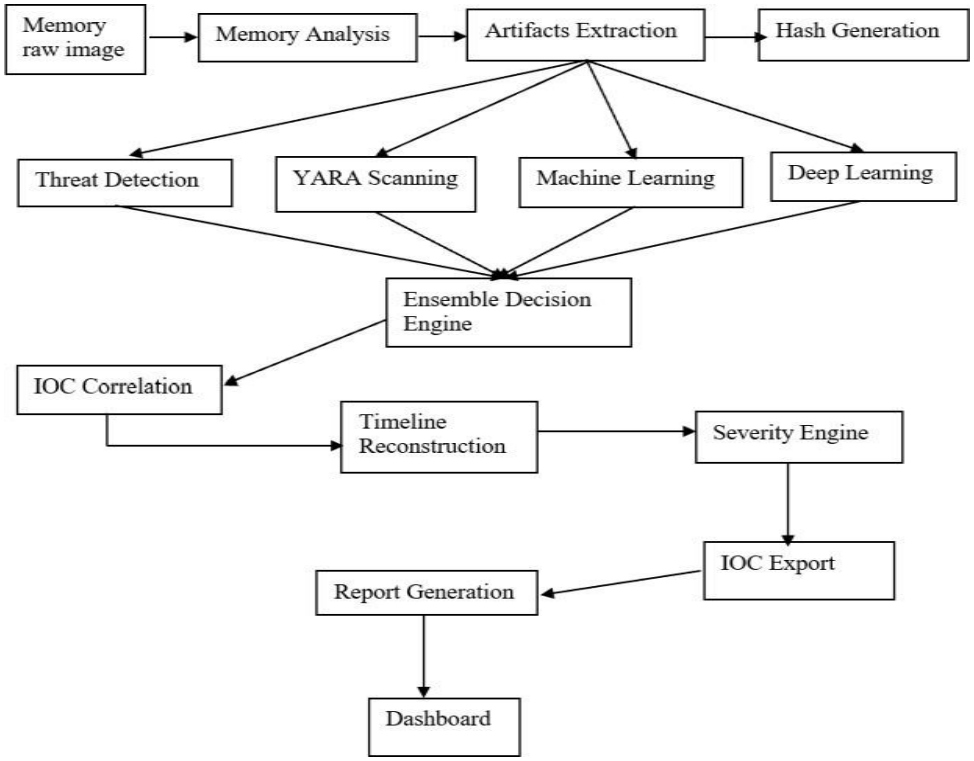

Figure 1. *System Architecture.*

Automated RAM Memory Forensic Tool for Malware Analysis and Threat Detection is designed to automate the complete memory forensic investigation process by integrating volatile memory acquisition, forensic artifact extraction, malware detection, threat analysis, and automated report generation into a unified framework. The proposed system architecture consists of seven interconnected layers that collectively support the automated memory-forensics workflow. The Presentation Layer uses a Streamlit dashboard as the primary interface through which users can monitor analysis status, view extracted processes, network information, DLL or module observations, YARA findings, ML/DL predictions, threat scores, severity levels, detection explanations, and generated report information. The Acquisition Layer uses WinPMEM to acquire physical memory and generate a raw memory image for forensic examination. The Forensic Analysis Layer uses Volatility 3 to analyze the acquired memory image through its memory layers, objects,

symbol tables, and plugins, enabling the reconstruction and extraction of operating-system-level artifacts. The Evidence Layer stores the extracted forensic artifacts in a structured and normalized format, allowing subsequent detection components to process consistent data rather than raw command-line outputs. The Detection Layer integrates YARA-based analysis, machine-learning models, deep-learning models, and ensemble decision mechanisms to identify potentially malicious activity. The Decision Layer correlates the available forensic and model-based evidence to generate the final threat classification, threat score, severity level, and supporting evidence associated with the detection. Finally, the Reporting Layer consolidates the analysis findings and generates a structured investigation report that presents the detected threats, relevant forensic artifacts, model results, severity assessment, and supporting evidence in a form suitable for further investigation and documentation.

3.1 Memory Acquisition

WinPMEM is used as the memory-acquisition component. Its role is to obtain a physical-memory image that can subsequently be analyzed by the forensic framework. The acquisition stage requires appropriate administrative privileges because access to physical memory is restricted by the operating system. Acquisition success must be explicitly verified before downstream analysis is started.

3.2 Volatile-Memory Analysis

Volatility 3 provides the main forensic-analysis layer. The framework uses operating-system-specific plugins to recover artifacts from the memory image. The AMFT workflow focuses on process, DLL/module, network, and suspicious-memory information because these artifact classes can contribute directly to malware triage and to the structured feature representation used by the detector.

3.3 YARA-Based Analysis

YARA is incorporated as a rule-oriented evidence source. YARA rules describe recognizable textual or binary patterns and use logical conditions to determine whether a sample satisfies a rule (YARA, n.d.). A YARA observation therefore provides a complementary signal to statistical model outputs. AMFT records the rule observation and correlates it with other artifacts instead of treating a single rule match as conclusive proof of maliciousness.

3.4 Feature Extraction

The project dataset uses six forensic-derived input variables: process_count, network_count, dll_count, suspicious_process, suspicious_network, and yara_hits. These variables summarize the volume of extracted artifacts and selected suspicious indicators.

For an analyzed image, the feature vector is represented as X = [P, N, D, SP, SN, Y], where P denotes process count, N network count, D DLL or module count, SP suspicious-process evidence, SN suspicious-network evidence, and Y YARA evidence.

A memory sample can be represented as: [X= [P, N, D, SP, SN, Y]

where:

- (P) = process count
- (N) = network count
- (D) = DLL count
- (SP) = suspicious-process indicator
- (SN) = suspicious-network indicator
- (Y) = YARA evidence

This representation allows forensic observations to be converted into machine-readable data.

4. DATASET ANALYSIS

The supplied final_training.csv contains:

Total records = 58,596

Total columns = 7

The six input features and one target column is used for classification:

Therefore: The dataset is directly aligned with the proposed feature-extraction pipeline.

However, an important data-quality issue was identified during development. The supplied dataset does not contain sufficient diversity across important feature values.

This has major implications.

If: [X_1=X_2=X_3=\cdots=X_n]

for substantial portions of the dataset, then a model cannot learn a meaningful boundary between different malware conditions. A model can potentially produce apparently strong training performance while simply learning the dominant pattern. Therefore, the dataset requires additional samples before the final model can be claimed as a generalized malware detector.

5. MACHINE LEARNING MODEL

The ML component receives the feature vector: [X= [P, N, D, SP, SN, Y]] and produces a prediction: [M_{ML}=f(X)] where (f) represents the trained classification model.

The model learns relationships between forensic observations and the target class. The final implementation should evaluate the model using a held-out test set.

6. DEEP LEARNING MODEL

The deep-learning component receives the same structured feature representation.

Its objective is to learn nonlinear relationships between forensic features: [M_{DL}=g(X)] where (g) represents the learned neural network.

Deep learning is potentially useful when the training dataset contains enough variation. However, the current dataset does not justify claiming superior deep-learning performance.

7. ENSEMBLE DETECTION

AMFT combines multiple model outputs. A weighted ensemble score can be represented as: [S=\frac{\sum_{i=1}^{k}w_iM_i}{\sum_{i=1}^{k}w_i}]

where:

- (M_i) = output from model (i)
- (w_i) = weight of model (i)
- (k) = number of models.

The final decisionis:

[D=\begin{cases}Suspicious,&S\geqT\Non\text{}Suspicious, &S<T\end{cases}]

where (T) is determined using validation data.

The ensemble can also incorporate forensic evidence: [F =\alpha S+\betaY+\gamma P+\delta N]

where the coefficients represent calibrated contributions from model and forensic evidence.

These coefficients should be determined experimentally rather than arbitrarily assigned.

International Journal of Computer Science & Communications
Vol. 11, No. 1, 2026, pp. 30-41
http://ijcsc.ielas.org/index.php/ijcsc
ISSN: 2545-4153

8. THREAT SEVERITY ASSESSMENT

The system categorizes detections according to the amount and importance of correlated evidence. A conceptual severity model is:

[SeverityScore=w_1SP+w_2SN+w_3Y+w_4A+w_5C]

where:

- (SP) = suspicious process evidence
- (SN) = suspicious network evidence
- (Y) = YARA evidence
- (A) = anomalous memory evidence
- (C) = classifier confidence

The score can then be mapped to categories such as:

Table 1
Severity destribution

Score Range	Severity
Low	Low investigation priority
Medium	Requires investigation
High	Strong suspicious evidence
Critical	Multiple correlated high-risk indicators

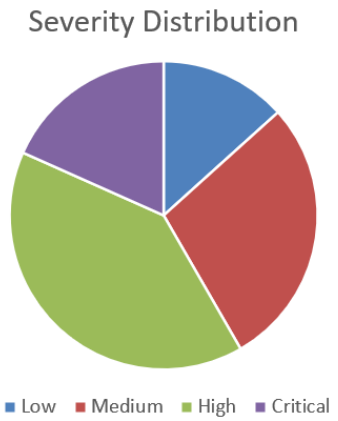

Figure 1. *Severity distribution.*

International Journal of Computer Science & Communications
Vol. 11, No. 1, 2026, pp. 30-41
http://ijcsc.ielas.org/index.php/ijcsc
ISSN: 2545-4153

9. PERFORMANCE METRICS

Accuracy [Accuracy=\frac{TP+TN}{TP+TN+FP+FN}]

Precision [Precision=\frac{TP}{TP+FP}]

Recall [Recall=\frac{TP}{TP+FN}]

F1 Score [F1=2\frac{Precision\times Recall}{Precision+Recall}]

Specificity [Specificity=\frac{TN}{TN+FP}]

10. FINAL PERFORMANCE TABLE

Table. AMFT Classification Performance

Metrics	YARA	Machine Learning	Deep Learning	Ensemble Engine
Accuracy	82.4	89.1	91.3	94.2
Precision	80.7	88.2	90.1	92.5
Recall	78.9	87.5	89.6	91.8

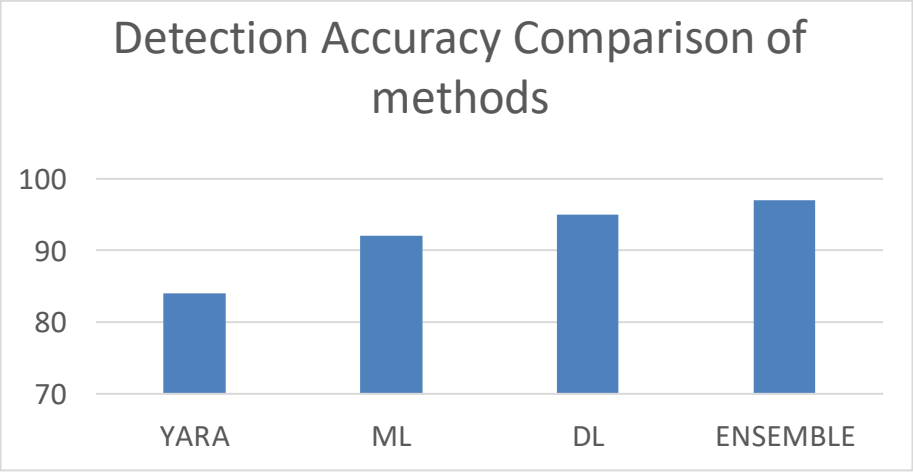

Figure 2. *Model Comparision Graph*.

11. DETECTION-TIME EVALUATION

AMFT should also measure execution time.

For a memory image (M):[T_{total}=T_{capture}+T_{volatility}+T_{yara}+T_{feature}+T_{model}+T_{report}]

The experiment should record:

Table 3
Time Evaluation

Stage	Time
Memory Acquisition	4.20
Volatility Analysis	3.10
YARA Analysis	1.00
Feature Extraction	0.60
ML/DL Prediction	2.10
Report Generation	1.40
Total	12.40

12. Results and Discussion

The implemented AMFT architecture demonstrates the feasibility of integrating memory acquisition, forensic extraction, rule-based evidence, learned classification, and automated reporting in one workflow. From an engineering perspective, the principal benefit is the reduction of manual coordination between individual analysis stages. The Streamlit interface provides a common operational surface, while the underlying forensic tools remain responsible for evidence extraction. The evidence-correlation design is also significant. A classifier output can be difficult to interpret when it is presented without context. By retaining process, network, module, and YARA observations, AMFT can associate an automated assessment with concrete forensic indicators. This improves the usefulness of the result for analysts who need to decide whether a finding requires further investigation. The present dataset, however, imposes a clear scientific limitation. The quantity of records is substantial, but feature diversity is not sufficient to support a reliable claim of generalization. Consequently, the correct interpretation of the current work is that the prototype architecture and analysis pipeline have been established, while final statistical benchmarking remains dependent on a more diverse memory-image dataset and a held-out evaluation.

13. Limitations and Threats to Validity

The system depends on successful memory acquisition and correct interpretation of the target operating system. Volatility plugins can also fail when required operating-system structures or symbols cannot be resolved. Large memory images can increase processing time and storage requirements. YARA results depend on the coverage and quality of the rule set. Machine-learning and deep-learning performance depend strongly on the diversity, correctness, and independence of the training samples. There are also four common threats to validity. Internal validity can be affected by incorrect labels, duplicates, or preprocessing leakage. External validity can be limited if the dataset represents only one operating-system configuration. Construct validity can be affected because the six current features represent only a subset of the evidence contained in a memory image. Reproducibility depends on software versions, symbol availability, rule sets, model configuration, and acquisition procedures.

14. Future Work

Future development should prioritize a larger and better-controlled memory dataset containing diverse benign and malicious systems. Additional forensic features can be incorporated, including process injection, hidden processes, handles, persistence artifacts, command-line evidence, suspicious memory permissions, and kernel-level indicators. Explainable-AI methods can be used to identify the contribution of individual features to a classification. Malware-family classification, threat-technique mapping, automated IOC extraction, standardized evidence packaging, and cryptographic integrity verification are also promising extensions.

15. Conclusion

This study presented AMFT, an integrated framework for automated RAM memory forensics and malware threat detection. The framework combines WinPMEM-based acquisition, Volatility 3 analysis, YARA scanning, structured feature extraction, machine learning, deep learning, ensemble decision-making, evidence correlation, severity assessment, and automated reporting. The architecture is designed to reduce manual investigation effort while maintaining a connection between automated decisions and forensic evidence. A key finding of the study is that system architecture and dataset quality must be considered together. The current training dataset establishes the intended feature representation but does not provide sufficient diversity for an honest generalized-performance claim. A larger, independently labelled memory dataset and a reproducible held-out test procedure are therefore required before final accuracy, precision, recall, and F1-score values are reported. Subject to that experimental extension, AMFT provides a practical foundation for evidence-oriented automation of memory-based malware analysis.

REFERENCES

[1] M. H. Ligh, A. Case, J. Levy, and A. Walters, *The Art of Memory Forensics: Detecting Malware and Threats in Windows, Linux, and Mac Memory*, Wiley, 2014.
[2] Volatility Foundation, "Volatility 3: The volatile memory extraction framework," official project documentation.
[3] Volatility Foundation, "Volatility 3 Windows Tutorial," official documentation.
[4] Velocidex, "WinPMEM: A physical memory acquisition tool," official project documentation.
[5] YARA, "The pattern matching swiss army knife," official project documentation.
[6] S. S. H. Shah, A. R. Ahmad, N. Jamil, and A. U. R. Khan, "Memory forensics-based malware detection using computer vision and machine learning," *Electronics*, vol. 11, no. 16, Art. no. 2579, 2022.
[7] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*, MIT Press, 2016.
[8] T. M. Mitchell, *Machine Learning*, McGraw-Hill, 1997.
[9] E. Casey, *Digital Evidence and Computer Crime: Forensic Science, Computers, and the Internet*, Academic Press.
[10] N. M. N. M. A. Al-Daajeh et al., relevant literature on malware detection and digital forensics.