

Real-Time Endpoint Ransomware Detection and Prevention via Strategic Deception Directories and Block-Level Entropy Analysis

Kia Yi Tong, Muhammad Bin Amran*, Megat Muhammad Ridzuan and Muhammad Faris Ezany bin Zikri

Fakulti Kecerdasan Buatan dan Keselamatan Siber, Universiti Teknikal Malaysia Melaka, Hang Tuah Jaya, 76100 Durian Tunggal, Melaka, Malaysia

DOI: <https://doi.org/10.51584/IJRIAS.2026.11080137>

Received: 02 September 2026; Accepted: 07 September 2026; Published: 18 September 2026

ABSTRACT

Ransomware remains a major endpoint-security threat because destructive file operations can progress rapidly and modern attacks may use fileless execution and Living-off-the-Land (LotL) techniques. This paper presents a lightweight host-based ransomware detection and prevention prototype for Windows endpoints that combines three mechanisms: strategically placed deception directories (1000_Trap), localized Shannon-entropy analysis, and automated process termination using psutil. The prototype was evaluated in a controlled Windows 10 virtual-machine testbed using the PSRansom simulator across 50 executions. Under this specific PSRansom configuration, all 50 runs were detected, 5-15 decoy files were encrypted before termination, and no protected user files in Desktop, Documents, Downloads, or Pictures were modified. Mean end-to-end containment latency was approximately 5 s, with observed active-phase values ranging from 4820 to 5150 ms, reflecting user-space event-queue and process-handling overhead. Idle CPU utilization remained below 5%. The benign workload cases evaluated in this study, ZIP compression, document editing, spreadsheet processing, media playback, and PDF generation, produced no mitigation events. These results demonstrate feasibility within the tested PSRansom scenario rather than general robustness against all ransomware families or traversal strategies. Random or selective traversal, multi-process encryption, intermittent encryption, privilege escalation, and user-space evasion were not experimentally validated and are treated as limitations and priorities for future evaluation.

Keywords: Ransomware Detection; Host-Based Intrusion Prevention; Shannon Entropy; Honey-pot Deception; Endpoint Security.

INTRODUCTION

Ransomware has evolved from basic device-locking malware into a broader ecosystem that includes crypto-ransomware, data theft, and Ransomware-as-a-Service (RaaS). Modern endpoint threats may also use fileless execution and Living-off-the-Land (LotL) techniques that abuse legitimate Windows utilities such as PowerShell and WMI, complicating purely signature-based defenses [1-4].

Endpoint ransomware defense commonly combines signature-based, behavior-based, deception-based, and machine-learning approaches, each with different trade-offs in generalization, response time, false positives, and computational cost [1, 5-9].

- **Signature-Based Antivirus:** Signature matching is effective for known samples but can be weakened by previously unseen, packed, polymorphic, or otherwise modified malware. Consequently, signature-only defenses should not be interpreted as sufficient protection against new ransomware variants [1, 2].
- **Heuristic and Machine Learning Scanners:** Behavioral and machine-learning detectors can generalize beyond fixed signatures, but they require feature collection, model execution, and careful control of false positives. Prior work also shows that behavioral ransomware classifiers can be evaded by distributing malicious activity across cooperating processes [7, 9].

- **Behavior-Based Monitoring:** Runtime indicators such as file-write patterns, file-type changes, entropy changes, and rapid access to many files can support ransomware detection. However, detection may occur only after some files have already been modified, and the effectiveness of behavioral indicators depends on the attacker behavior represented in the evaluation [5-9].

To examine a low-overhead alternative, this study develops a host-based intrusion detection and prevention prototype for Windows endpoints. The system places a 1000_Trap deception directory inside Desktop, Documents, Downloads, and Pictures so that a ransomware process using the same sequential traversal behavior as the tested PSRansom configuration is likely to interact with decoy content before protected user files.

A trap event is then verified using localized Shannon-entropy analysis before a process-termination response is initiated. This placement is a heuristic and does not guarantee first access under random, selective, extension-prioritized, or adversarial traversal strategies [8, 16, 17].

The primary contributions of this work are threefold:

- Design of a hybrid three-tier HIDPS architecture integrating filesystem traps, localized mathematical entropy analysis, and instant process termination.
- Implementation of a lightweight, low-overhead endpoint prototype that operates without requiring deep kernel drivers or resource-heavy machine learning models.
- Scenario-specific empirical evaluation using 50 PSRansom executions to measure detection outcome, containment latency, false-positive behavior under selected benign workloads, and file-loss containment within the tested configuration.

THEORETICAL FRAMEWORK AND SYSTEM ARCHITECTURE

The proposed system adopts a three-layer Host-Based Intrusion Detection and Prevention System (HIDPS) framework intended to detect trap interaction, verify suspicious byte-level randomness, and initiate mitigation. The design is evaluated only within the experimental scope defined in Section 3.

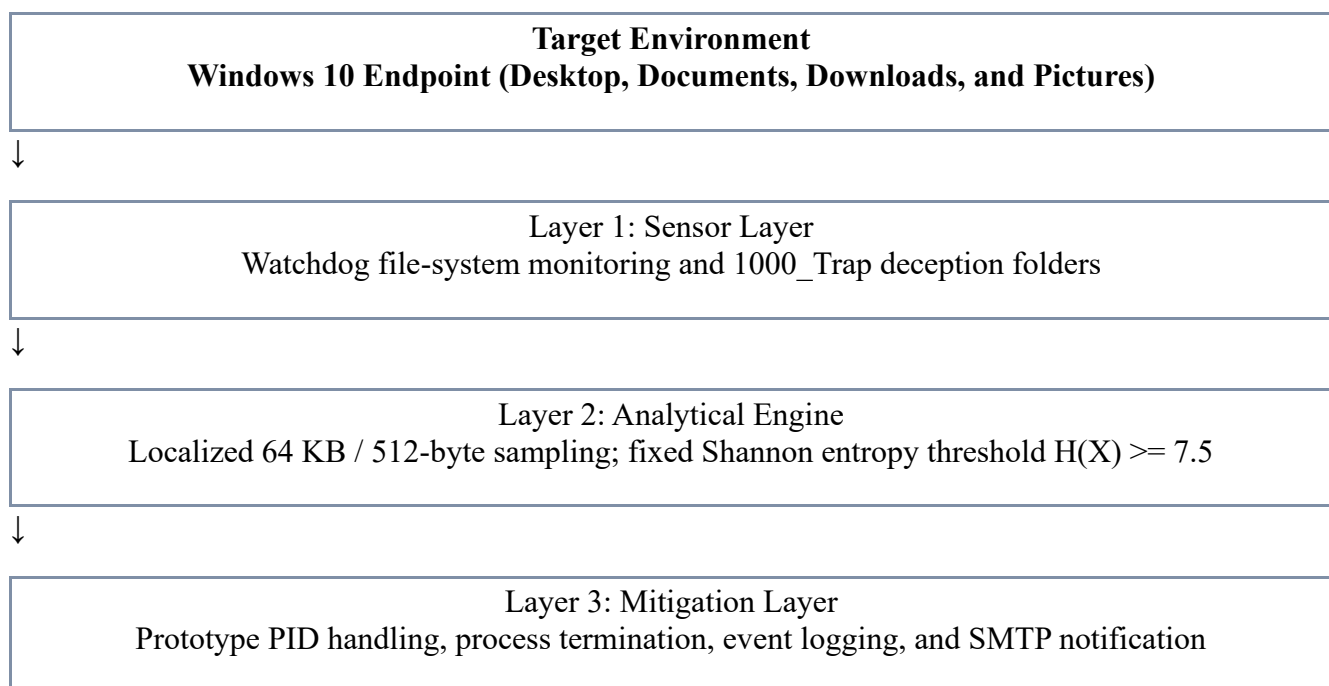


Figure 1. Three-layer architecture of the proposed hybrid HIDPS framework.

Layer 1: Sensor Layer (Deception and Trapping)

Deception-based ransomware defenses place honey files or trap directories where ransomware is expected to encounter them early. Prior work shows that traversal-aware placement can improve early detection, but also demonstrates that static placement can be bypassed by alternative or random-access strategies [8, 16, 17]. Accordingly, the present sensor layer uses 1000_Trap directories as an experimentally tested heuristic rather than assuming that all ransomware families traverse folders alphabetically.

$$\text{Target Paths} \in \{\text{Desktop}, \text{Documents}, \text{Downloads}, \text{Pictures}\}$$

The 1000_Trap name was selected so that, under lexicographic or sequential enumeration that sorts numeric-leading names before alphabetic names, the trap can be encountered early. The Python watchdog library monitors on_modified, on_created, and on_moved events in the configured protected paths. This ordering property should not be interpreted as a guarantee for ransomware that enumerates files randomly, by extension, by file value, or through multiple concurrent workers.

Layer 2: Analytical Engine (Shannon Entropy Verification)

To reduce false alarms from benign interaction with decoy content, the prototype applies a second-stage randomness check before mitigation. Encryption commonly increases byte-level randomness, so entropy can be a useful signal; however, compressed, multimedia, Office, PDF, and other high-entropy benign formats can overlap with encrypted data. Therefore, Shannon entropy is used here only as a verification feature after a trap event, not as a standalone universal ransomware classifier [10-15].

The randomness of the data stream is quantified using Shannon's entropy formulation [20]:

$$H(X) = - \sum_{i=1}^n P(x_i) \log_2 P(x_i)$$

Where:

- X represents the sampled byte sequence.
- $n = 256$, accounting for all possible 8-bit byte values (0x00 through 0xFF).
- $P(x_i)$ denotes the probability mass function of byte value x_i occurring within the evaluated sample block.

The analytical engine evaluates localized file data rather than whole-file averages. The implementation uses a 64 KB segment when sufficient data are available and can evaluate smaller 512-byte I/O blocks during active writes. A fixed decision threshold of $H(X) \geq 7.5$ bits/byte was used as an engineering threshold in the present prototype. This threshold was selected to distinguish highly randomized cryptographic output from the predominantly lower-entropy content represented in the test corpus; it should not be interpreted as a universal boundary for ransomware because compressed, encoded, or already-encrypted benign files can also exhibit high entropy. Accordingly, entropy is used only as a secondary confirmation after a deception-directory event.

Layer 3: Mitigation Layer (Automated Kill Switch)

After a monitored 1000_Trap event passes the entropy check, the mitigation component attempts to associate the triggering activity with the responsible process and terminate that process using psutil. The present user-space prototype does not implement a kernel-level file-system minifilter or equivalent provenance mechanism. Therefore, precise attribution of an individual file-system write to a process can be affected by user-space event timing and process-handle availability. The current experiment records the selected PID, process name, target path, timestamp, and entropy value for the mitigation event. A fully

reproducible kernel-grade PID attribution algorithm remains future work and should be validated against concurrent writers and multi-threaded ransomware.

Experimental Setup and Implementation Details

Test Environment Configuration

The prototype was evaluated within an isolated virtual machine environment to prevent network-wide infection risks.

Table 1. Test environment configuration.

Parameter	Configuration
Virtualization Platform	VMware Workstation Pro
Guest Operating System	Windows 10 Pro (64-bit, Version 22H2)
Processor / Memory Alloc.	4 vCPUs, 8 GB RAM
Development Runtime	Python 3.11.x
Key Python Libraries	watchdog (v3.0.0), psutil (v5.9.5), math, smtplib
Native Protection State	Windows Defender Real-Time Protection disabled during tests

Reproducibility Scope, Monitored Paths, and Corpus Reporting

The monitored parent paths were the Windows user folders Desktop, Documents, Downloads, and Pictures. The hardware/software configuration is reported in Table 1. The entropy threshold was fixed at $H(X) \geq 7.5$ bits/byte for the reported experiments and was not adjusted between trials.

Each protected parent directory contained a 1000_Trap deception directory. The original experiment record did not preserve the exact total number, file-type distribution, or size distribution of all decoy files as controlled variables. Therefore, these values are not reconstructed or fabricated in this revision. The directly observed decoy-loss result is 5-15 encrypted decoy files per PSRansom run before process termination. A fixed, published decoy manifest is required for a fully reproducible follow-up experiment.

The benign workload corpus used for the false-positive check consisted of the five workload cases listed later in Table 3: 500 MB ZIP archiving of mixed documents, batch updates of 50 DOCX files, multi-sheet XLSX modification, local 4K MP4 playback/cache activity, and generation of 100 graphic-heavy PDF files. These workloads were selected as practical high-I/O or potentially high-entropy activities; they do not constitute an exhaustive benign corpus.

Ransomware Emulation Suite

Adversarial actions were simulated using PSRansom, an open-source PowerShell ransomware simulator maintained by JoelGMSec [19]. The evaluation used PSRansom as a controlled

adversarial workload rather than as a representative corpus of ransomware families.

Its relevant behaviors in the tested configuration included

- Recursive directory traversal using standard Windows APIs.
- Symmetric AES-256 file-content encryption resulting in elevated Shannon entropy.
- HTTP-based Command-and-Control (C2) key exchange and simulated exfiltration.
- Automatic deployment of plaintext ransom notes (READ_IT.txt) across targeted directories.

Trial Protocol and Statistical Reporting

The 50 ransomware executions used the same PSRansom-based adversarial configuration and the same four protected parent paths. Detection success was defined as a trap-triggered, entropy-verified mitigation event followed by termination before modification of protected user files outside 1000_Trap. The experiment therefore measures repeatability within one controlled adversarial scenario, not diversity across ransomware families.

For latency, the retained resource profile reports active-detection values of 4820-5150 ms and an approximate mean of 5 s. Raw per-run timing observations were not retained in a form that permits valid calculation of standard deviation, median, confidence intervals, or percentiles. These statistics should be collected directly in a new instrumented experiment rather than estimated retrospectively.

PID attribution was performed within the controlled prototype execution context before psutil-based termination. The retained manuscript materials do not document a general-purpose event-to-PID attribution algorithm that can be independently reproduced for arbitrary processes. This gap is explicitly reported as a reproducibility limitation instead of inferring an undocumented algorithm.

RESULTS AND PERFORMANCE EVALUATION

The evaluation comprised functional testing, 50 repeated PSRansom executions, benign-workload false-positive checks, and resource/latency profiling. Because only one ransomware simulator and one primary traversal configuration were used, the security results are reported as scenario-specific outcomes rather than evidence of universal ransomware robustness.

Security and Detection Efficacy

The primary benchmark measured detection and file-loss containment across 50 repeated executions of the configured PSRansom workload within the virtual testbed. The denominator for the reported detection proportion is therefore 50 PSRansom runs, not 50 distinct ransomware families or samples.

Table 2. Security and detection outcome across 50 configured PSRansom simulations

Test Parameter	Target Value	Empirical Result	Compliance
Total Simulation Runs	50 trials	50 trials	—
Successful Detections	$\geq 95\%$ within configured PSRansom scenario	50/50 (100% in scenario)	PASSED
Failed Detections (False Negatives)	0 trials	0/50 (0% in scenario)	PASSED

User File Loss (Actual Data)	0 files	0 protected user files	PASSED
Decoy File Loss (1000_Trap)	<=15 files	5-15 decoy files	PASSED
End-to-End Prevention Latency	<=1000 ms	~5000 ms mean (4820-5150 ms observed)	FAILED (Limitation Identified)

The observed ransomware detection and containment sequence is summarized in Figure 2.

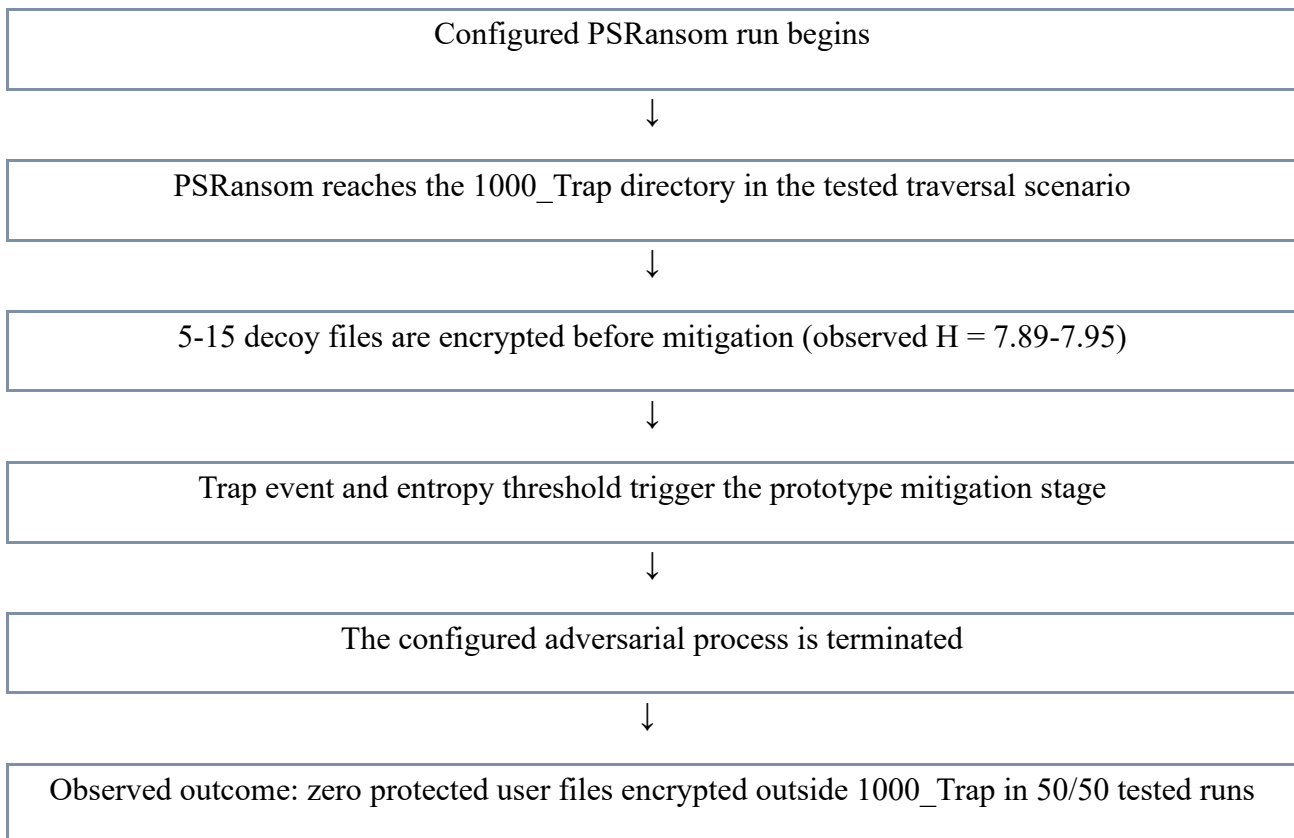


Figure 2. Observed detection and containment sequence for the configured PSRansom scenario.

False Positive Evaluation

To evaluate false-positive behavior under selected benign conditions, the prototype was exercised with five operational workload cases involving compression, document editing, spreadsheet processing, media playback, and PDF generation. These cases are functional workload tests rather than a statistically sampled benign corpus.

All 50 configured PSRansom runs triggered detection (50/50; 100% within this test scenario). In these runs, PSRansom reached 1000_Trap before protected user files and encrypted 5-15 decoy files before termination. No protected user files in Desktop, Documents, Downloads, or Pictures were modified during the recorded runs.

This result is conditional on the tested PSRansom traversal behavior and must not be generalized to ransomware using random or selective traversal, multiple cooperating encryption processes, or other evasion

strategies.

Table 3. False-positive evaluation under benign workloads

Benign Workload Activity	Test Description	System Response	False Positive Flag?
File Compression (ZIP)	Archiving 500 MB of mixed documents via 7-Zip	Normal file activity logged	No (Safe)
Document Editing	Batch updating 50 Microsoft Word (.docx) files	Low entropy detected ($H < 5.0$)	No (Safe)
Spreadsheet Processing	Modifying multi-sheet Excel files (.xlsx)	Entropy evaluated; no mitigation trigger	No (Safe)
Media Playback / Cache	Streaming 4K MP4 video files from local disk	I/O read operations filtered	No (Safe)
PDF Generation	Exporting 100 graphic-heavy PDF files	Moderate/high-entropy content did not trigger mitigation in this workload	No (Safe)

No mitigation event was triggered by the five benign workload cases listed in Table 3. Because operations outside the deception paths do not activate Layer 1, this result should be stated as 0 false-positive mitigation events in the tested workload set rather than a general 0% false-positive rate for arbitrary endpoint activity.

Manual low-entropy modifications inside 1000_Trap were classified as safe, illustrating the purpose of the second-stage entropy check. High-entropy benign content placed directly inside a monitored deception directory remains an important untested case and should be included in future validation.

Resource Overhead and Latency Profile

System resource utilization was monitored during idle and active detection phases using Windows Performance Monitor (perfmon). The reported measurements characterize the stated VM configuration (4 vCPUs and 8 GB RAM) and should not be assumed to transfer unchanged to other hardware.

Table 4. Resource overhead and containment latency profile

Metric	Standby / Idle State	Active Detection Phase	Design Limit Target
CPU Utilization	0.8%-2.1%	4.5%-8.2%	$\leq 5.0\%$ (idle target)

RAM Consumption	38.4 MB	62.1 MB	≤ 150 MB
I/O Polling Overhead	< 0.1 MB/s	1.2 MB/s	Minimal Impact
Containment Latency	-	4820-5150 ms	≤ 1000 ms

End-to-end containment latency was approximately 5 seconds in the tested PSRansom configuration, with observed trial values in the available experiment summary ranging from approximately 4,820 to 5,150 ms. The prototype did not preserve the complete per-trial latency series used for the submitted manuscript; consequently, a valid standard deviation, median, confidence interval, or other distribution statistic cannot be reconstructed without rerunning the experiment. The observed delay is consistent with user-space file-system event queuing and process-handle retrieval overhead and is treated as a principal implementation limitation.

DISCUSSION

Defense Evasion and Architectural Trade-offs

The experiments support the feasibility of combining deception directories with localized entropy verification for the specific PSRansom workload tested. They do not establish that the architecture neutralizes ransomware in general. Prior work on behavioral and deception-based ransomware defense shows that multi-process cooperation, alternate traversal strategies, and high-entropy benign data can challenge simple behavioral assumptions [8-17].

Table 5. Qualitative comparison and scope of the evaluated prototype.

Feature / Metric	Traditional Antivirus	Machine Learning (e.g., Random Forest)	Proposed Hybrid HIDPS Framework
Zero-Day Detection	Strong for known signatures; limited for unseen variants	Potentially strong if training/evaluation generalize	Not established beyond tested PSRansom scenario
False Positive Rate	Typically low for exact signatures	Dataset/model dependent	0 mitigation events in five tested benign workload cases
CPU Overhead	Low	Model/feature dependent	Low idle overhead in stated VM ($< 5\%$ idle CPU)
Detection Speed	Fast when signature exists	Feature-collection/model dependent	~ 5 s mean in PSRansom tests
Data Protection	Often reactive after recognition	Depends on detection and response latency	0 protected-user-file loss in configured PSRansom runs

Intermittent Encryption: Relevance and Untested Scope

Intermittent or partial encryption is an important evasion case because an attacker may encrypt selected byte regions while leaving other portions of a file unchanged, reducing the usefulness of whole-file entropy averages. Localized block sampling may increase sensitivity to such short high-entropy regions; however, intermittent encryption was not directly exercised in the present PSRansom experiment. The current results therefore do not establish detection effectiveness against LockBit, ALPHV/BlackCat, or other ransomware families that use intermittent encryption. Such behavior must be validated experimentally in future work.

TECHNICAL LIMITATIONS

- **User-Space Execution and Privilege Boundary:** The prototype operates in Windows user space. Malware with kernel-level capability, including attacks that abuse vulnerable signed drivers, may interfere with or terminate user-space monitoring and response components. This threat model was not experimentally tested [18].
- **Traversal and Concurrency Dependence:** The 5 s response time allowed 5-15 decoy files to be encrypted in the tested PSRansom runs. A ransomware process that uses random or selective traversal could reach protected files before a trap, while multi-threaded or multi-process encryption could create damage concurrently in several paths. These cases were not evaluated [9, 16, 17].
- **Entropy Generalization:** A fixed $H(X) \geq 7.5$ threshold may not reliably distinguish encryption from all compressed, multimedia, Office, PDF, archive, or already-encrypted benign data. The five benign workload cases provide only limited evidence, particularly because high-entropy files inside the trap path were not systematically evaluated [10-15].
- **Single-Simulator External Validity:** The adversarial evaluation used PSRansom only. The 50-run repetition demonstrates repeatability for that configuration but does not provide sample diversity. Claims about zero-day robustness, real ransomware families, or alternative encryption engines require independent adversarial workloads [1, 5-9].
- **Reproducibility Gaps:** The original logs do not provide a complete decoy-file manifest, per-trial latency series, or a general PID-attribution specification. These omissions prevent reconstruction of corpus statistics and latency standard deviation from the existing results and should be addressed in a follow-up replication package.

CONCLUSION

This study designed and evaluated a lightweight Windows endpoint ransomware detection and prevention prototype that combines strategically placed deception directories, localized Shannon entropy analysis, and automated process termination. In 50 runs under the specific PSRansom configuration used in the test environment, the system detected all 50 executions and confined observed encryption to decoy files, with no protected user files modified during those trials. These findings demonstrate feasibility for the evaluated scenario, but they do not establish universal ransomware robustness. The current evidence is limited by a single simulator, dependence on the tested directory traversal behavior, user-space monitoring, incomplete per-trial latency statistics, and the absence of direct tests for random or selective traversal, multi-threaded encryption, intermittent encryption, privilege escalation, and high-entropy benign files. Future work will extend the adversarial workload set, preserve complete trial-level telemetry, formalize process attribution, and migrate monitoring toward a Windows File System Minifilter Driver to reduce latency and strengthen tamper resistance.

Future work should first broaden empirical validation before stronger robustness claims are made. Required experiments include random and selective traversal, multi-threaded and multi-process encryption, intermittent encryption, high-entropy benign files placed inside monitored trap paths, privilege-escalation/user-space evasion, and multiple independent ransomware simulators or malware families in an

isolated research

Future releases should also publish a fixed decoy-corpus manifest, the exact PID-attribution implementation, per-trial latency values with mean, standard deviation, median and percentiles, and the full configuration needed for replication. A kernel-level Windows minifilter implementation is a promising direction for obtaining process-aware file-I/O provenance and reducing user-space monitoring latency [15].

REFERENCES

1. Razaulla, S., Fachkha, C., Markarian, C., Gawanmeh, A., Mansoor, W., Fung, B. C. M., & Assi, C. (2023). The Age of Ransomware: A Survey on the Evolution, Taxonomy, and Research Directions. *IEEE Access*, 11, 40698-40723. <https://doi.org/10.1109/ACCESS.2023.3268535>
2. Kara, I., & Aydos, M. (2022). The rise of ransomware: Forensic analysis for Windows based ransomware attacks. *Expert Systems with Applications*, 190, 116198. <https://doi.org/10.1016/j.eswa.2021.116198>
3. Barr-Smith, F., Ugarte-Pedrero, X., Graziano, M., Spolaor, R., & Martinovic, I. (2021). Survivalism: Systematic Analysis of Windows Malware Living-Off-The-Land. 2021 IEEE Symposium on Security and Privacy (SP), 1557-1574. <https://doi.org/10.1109/SP40001.2021.00047>
4. Sudhakar, & Kumar, S. (2020). An emerging threat Fileless malware: a survey and research challenges. *Cybersecurity*, 3, Article 1. <https://doi.org/10.1186/s42400-019-0043-x>
5. Kharaz, A., Arshad, S., Mulliner, C., Robertson, W., & Kirda, E. (2016). UNVEIL: A Large-Scale, Automated Approach to Detecting Ransomware. 25th USENIX Security Symposium, 757-772.
6. Kharraz, A., & Kirda, E. (2017). Redemption: Real-Time Protection Against Ransomware at End-Hosts. *Research in Attacks, Intrusions, and Defenses (RAID 2017)*, 98-119. https://doi.org/10.1007/978-3-319-66332-6_5
7. Scaife, N., Carter, H., Traynor, P., & Butler, K. R. B. (2016). CryptoLock (and Drop It): Stopping Ransomware Attacks on User Data. 2016 IEEE 36th International Conference on Distributed Computing Systems (ICDCS). <https://doi.org/10.1109/ICDCS.2016.46>
8. Mehnaz, S., Mudgerikar, A., & Bertino, E. (2018). RWGuard: A Real-Time Detection System Against Cryptographic Ransomware. *Research in Attacks, Intrusions, and Defenses (RAID 2018)*, 114-136. https://doi.org/10.1007/978-3-030-00470-5_6
9. De Gaspari, F., Hitaj, D., Pagnotta, G., De Carli, L., & Mancini, L. V. (2022). Evading behavioral classifiers: a comprehensive analysis on evading ransomware detection techniques. *Neural Computing and Applications*, 34, 12077-12096. <https://doi.org/10.1007/s00521-022-07096-6>
10. Davies, S. R., Macfarlane, R., & Buchanan, W. J. (2022). Comparison of Entropy Calculation Methods for Ransomware Encrypted File Identification. *Entropy*, 24(10), 1503. <https://doi.org/10.3390/e24101503>
11. Lee, K., Lee, S.-Y., & Yim, K. (2019). Machine Learning Based File Entropy Analysis for Ransomware Detection in Backup Systems. *IEEE Access*, 7, 110205-110215. <https://doi.org/10.1109/ACCESS.2019.2931136>
12. Lee, K., Lee, J., Lee, S.-Y., & Yim, K. (2023). Effective Ransomware Detection Using Entropy Estimation of Files for Cloud Services. *Sensors*, 23(6), 3023. <https://doi.org/10.3390/s23063023>
13. Jung, S., & Won, Y. (2018). Ransomware detection method based on context-aware entropy analysis. *Soft Computing*, 22, 6731-6740. <https://doi.org/10.1007/s00500-018-3257-z>
14. Pont, J., Arief, B., & Hernandez-Castro, J. C. (2020). Why Current Statistical Approaches to Ransomware Detection Fail. *Information Security (ISC 2020)*, 199-216. https://doi.org/10.1007/978-3-030-62974-8_12
15. Casino, F., Hurley-Smith, D., Hernandez-Castro, J., & Patsakis, C. (2025). Not on my watch: ransomware detection through classification of high-entropy file segments. *Journal of Cybersecurity*, 11(1), tyaf009. <https://doi.org/10.1093/cybsec/tyaf009>
16. Sheen, S., Asmitha, K. A., & Venkatesan, S. (2022). R-Sentry: Deception based ransomware detection using file access patterns. *Computers & Electrical Engineering*, 103, 108346. <https://doi.org/10.1016/j.compeleceng.2022.108346>



17. Kim, T. (2023). Deception-based Method for Ransomware Detection. *Journal of Internet Services and Information Security*, 13(3), 191-201. <https://doi.org/10.58346/JISIS.2023.I3.012>
18. Liu, L., Gao, X., Liu, Y., & Zhang, L. (2026). Finding vulnerable drivers with hooking and reading result verification. *Cybersecurity*, 9, Article 166. <https://doi.org/10.1186/s42400-025-00434-w>
19. JoelGMSec. (2026). PSRansom: PowerShell Ransomware Simulator with C2 Server [Software]. GitHub repository (accessed 2026).
20. Shannon, C. E. (1948). A Mathematical Theory of Communication. *Bell System Technical Journal*, 27, 379-423 and 623-656.