*Grenze International Journal of Engineering and Technology, January Issue*

# Hardware-Assisted Anomaly Detection in Storage Systems for Malware Resilience

Nazir Mohammed
ASP WEB SOLUTIONS LLC, India
Email: nazir.ms@gmail.com

***Abstract*— With the increasing sophistication of ransomware and other malicious software, traditional host-based security mechanisms are becoming ineffective due to their vulnerability to tampering. This paper introduces a novel storage- level anomaly detection framework that leverages independent hardware-assisted monitoring to track file system activity in real time. By integrating multi-layer data collection across network interfaces, storage controllers, and filesystem structures, the proposed system effectively distinguishes between normal and malicious disk operations. Experimental analysis of various benign applications and malware samples reveals distinct access patterns that can be used to detect emerging threats. The study highlights the potential for integrating low-level storage metrics into a robust, tamper-resistant security framework for next- generation secure data storage solutions.**

## I. INTRODUCTION

Ransomware has emerged as one of the most pervasive categories of cyber threats, characterized by its capability to encode digital content and demand payment in exchange for decryption. The advent and proliferation of Ransomware-as-a- Service (RaaS) platforms have significantly reduced the tech- nical threshold required to execute such attacks, empowering individuals with minimal cybersecurity expertise to deploy complex and highly effective malware. Contemporary strains of ransomware incorporate refined strategies such as concur- rent processing and selective encryption routines, allowing them to compromise vast datasets at unprecedented speeds. Certain variants have demonstrated the ability to lock tens of thousands of files in mere minutes.

Empirical assessments using large-scale file repositories have revealed that recent ransomware iterations are optimized for speed and efficiency, underscoring their growing sophis- tication and reinforcing the urgent demand for advanced and proactive detection frameworks. Traditional detection method- ologies typically rely on behavioral indicators gathered from within the affected system — including but not limited to system call traces, registry modifications, hardware usage metrics, and irregular communication patterns. Despite their usefulness, such host-based mechanisms remain susceptible to manipulation or circumvention, especially when the malware actively disables or bypasses onboard defense utilities.

Furthermore, delegating threat analysis to remote cloud infrastructures introduces additional challenges involving confidentiality, legal constraints, and compliance with regional data protection regulations. These drawbacks highlight the pressing requirement for a more robust, resilient, and tamper-resistant approach that operates independently of the host system and is immune to direct interference.

In response to this gap, this work introduces SHIELD — a novel architecture for secure, autonomous monitoring and profiling of low-level system activity aimed at identifying ransomware-induced anomalies.

*Grenze ID: 01.GIJET.12.1.160*

SHIELD leverages a multi- tiered storage hierarchy, gathering telemetry from layers such as the Network Block Device (NBD), Field-Programmable Gate Array (FPGA) interfaces, Serial ATA (SATA) Host Bus Adapters (HBAs), and underlying file system operations. This research outlines SHIELD's structural components, operational principles, and suitability for integration with machine learning classifiers designed to detect abnormal patterns in- dicative of ransomware attacks. Through empirical validation, this framework is shown to provide a reliable alternative to host-bound defenses, offering greater resilience and earlier threat identification capabilities.

## II. Related Work

Recent work in system-level monitoring and malware re- silience has shown an increased interest in low-level profiling mechanisms. Jyoti Singh[46] has developed multiple tools, including IHPP, which uses binary instrumentation to offer multi-mode profiling, aligning with our paper's emphasis on system granularity. Singh's[47] contributions also span into cybersecurity-focused Flask applications that highlight attack surface reduction and defensive logging—a relevant approach for hybrid monitoring. Additionally, Singh's[48] exploration of software transactional memory without hardware depen- dency provides architectural parallels to our hardware-assisted anomaly capture. Singh's[49] comparative simulation of open and closed-source development models also underlines the significance of monitoring behaviors under varied architectural strategies, which ties into our analysis of benign vs. malicious application behaviors.

Rizan[50] has contributed a series of advanced vision-based profiling techniques that parallel our work's emphasis on anomaly pattern detection. Rizan's[51] evaluation of object tracking algorithms under occlusion and lighting stress, along with edge detection using ant colony optimization, demon- strates the utility of pattern variance analysis—mirroring our FPGA-based metric deviation profiling. Rizan's[52] applica- tion of Mamba-based architectures for efficient image seg- mentation provides insights on computational efficiency, which could inspire future FPGA optimization. Recharla's[53] work in decentralized storage systems and memory partitioning also shares a foundational focus with our approach, par- ticularly in system-level architecture and dynamic resource allocation. Recharla[54] has also implemented parallel matrix operations and CI/CD pipelines in scalable cloud environ- ments. Recharla's[55] work also offers practical frameworks for extending our hardware-monitoring design to distributed systems. Srivastava[56] has investigated intrinsic motivation in reinforcement learning, offering a theoretical lens on self- driven exploration, which informs our paper's notion of au- tonomous anomaly profiling. Collectively, these works support the development of robust, independent security infrastructures and validate our pursuit of resilient storage-level anomaly detection.

## III. System Architecture and Methodology

### *A. Architectural Summary and Strategic Objectives*

The SHIELD framework has been conceptualized to deliver an affordable and platform-neutral infrastructure for monitor- ing low-level storage activities. By collecting telemetry from the Serial ATA (SATA) protocol layer, Network Block Device (NBD) communications, and underlying file system events, the system enables early-stage identification of anomalous or potentially malicious patterns in data behavior. It achieves this through unobtrusive and tamper-evident mechanisms, mini- mizing dependency on the host operating system's integrity.

- Decoupled Metric Acquisition Mechanism: This archi- tectural approach prioritizes autonomy from the operating environment by retrieving granular telemetry directly from hardware pathways, thereby bypassing compromised software stacks. It captures a diverse range of parameters, including transaction categories (read/write), data transfer lengths, and context-aware file system insights. Implementation utilizes the EXT4 format, where structural data such as superblock updates, allocation group metadata, inode changes, and data block manipulations are continuously monitored to establish a comprehensive behavioral profile.
- Deployment of an Open-Architecture SATA Host Con- troller: To circumvent the financial and licensing constraints posed by commercial SATA controller IPs, SHIELD integrates the open-source Groundhog SATA Host Bus Adapter, built upon FPGA technology. This open hardware solution supports direct interaction with SATA drives by utilizing Microsoft's SIRC protocol. The SHIELD infrastructure further extends the capabilities of this controller to support block-level disk provisioning, virtual mounting, and file system formatting through the NBD interface, ensuring seamless compatibility across a heterogeneous mix of operating systems.
- Reconfigurable Hardware Integration: For hardware- level interception and analysis, SHIELD employs the Digilent XUPV5 Field-Programmable Gate Array (FPGA) platform. Its inherent support for high-

throughput serial communication and flexible digital logic makes it well-suited for SATA-based applications. Key features of this implementation include:

- GTP Interfaces: High-bandwidth serial links enabling SATA Generation 2 signaling at up to 3 Gbps.
- Embedded SATA Interfaces: Dedicated hardware path- ways and connectors optimized for direct SATA protocol implementation.
- Programmable Logic Resources: Configurable memory blocks, arithmetic processing units, and logic cells are available for integrating advanced real-time detection engines in future revisions.

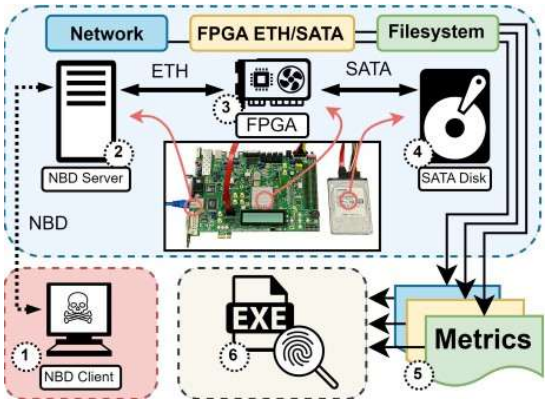


Fig. 1. EXT4 architecture with metrics used in ransomware detection

- Dual Support for Emulated and Native Storage: To accelerate the evaluation and debugging lifecycle, the ar- chitecture accommodates both software-based disk images and hardware storage media. This design flexibility ensures rapid prototyping during development stages and facilitates deployment in real-world setups. Although the presence of intermediary hardware can introduce marginal latency, future adaptations involving custom ASICs are expected to streamline performance and minimize bottlenecks.

*B. Deployment Framework and Operational Flow*

- End-to-End Execution Framework: The SHIELD op- erational lifecycle begins with an NBD-compatible client, representing either a physical system or an emulated instance, initiating communication over a network interface. This re- quest is relayed to the server component, which, in turn, synchronizes with the FPGA device. The FPGA captures op- erational metadata directly from the connected storage media. All actions are logged in advance of data persistence, allowing behavior-based detection algorithms to flag irregular sequences prior to disk write completion.
- Software Stack and Interface Components: The NBD daemon serves as the central middleware, orchestrating inter- actions between high-level file system calls and underlying hardware subsystems. It allows conventional storage proce- dures—such as mounting, writing, or reading data—while simultaneously harvesting information on usage frequency, byte-level access patterns, and temporal characteristics of disk transactions. This telemetry is correlated with low-level signals from the hardware for integrity validation and behavioral consistency checks.

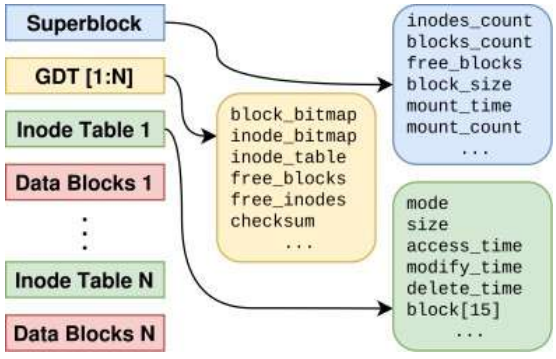


Fig. 2. EXT4 architecture with metrics used in ransomware detection

- FPGA-Driven Monitoring Infrastructure: Situated as an intermediary between storage interfaces and the data-handling software, the FPGA logic monitors I/O requests and associ- ated metadata. It captures operational commands (read/write), buffer sizes, timing markers, and structural mutations within the file system. This multi-angle data observation supports high-fidelity behavioral analysis by revealing deviations from standard access workflows.

The SHIELD methodology includes direct inspection of EXT4 file system structures using both buffer parsing and disk interrogation techniques. Noteworthy entities such as the superblock, group descriptors, and inode directories are closely monitored for unauthorized or unexpected changes. Figure 2 outlines the EXT4 metadata components that serve as indicators in the threat detection framework. Although journaling mechanisms and extent mapping structures exist within EXT4, their monitoring has not yet been incorporated into the current system build.

A twofold strategy governs data extraction within SHIELD:

(1) *proactive querying*, wherein SATA-level read operations allow detailed state inspection before any host-side alterations occur, and (2) *reactive parsing*, which involves real-time monitoring of memory buffers being accessed or modified by the operating system. This combined approach allows an exhaustive baseline scan at system initialization, followed by non-disruptive observation throughout subsequent activity.

## IV. Application Profiling Through Empirical Analysis

### *A. Testbed Configuration and Software Scope*

To evaluate SHIELD's efficacy in differentiating between non-malicious utilities and malicious software, empirical stud- ies were conducted using a curated set of ten ransomware families and ten legitimate software applications, detailed in Table I. The benign applications encompass a spectrum of disk-access behaviors, spanning multimedia processing tools, data compression utilities, file removal programs, and encryp- tion platforms.

The experiments utilized an isolated virtualized environment provisioned with 16 GB of volatile memory, an operating

Algorithm 1 Procedure for Recording Disk Activity Metrics via SHIELD

[1] Declare global structures for EXT4 metadata tracking

Initiate communication channel with SATA hardware Parse superblock starting from known disk offset EXT4 identifier is valid Load associated group descriptor entries descriptor entries Access and parse linked inode tables retrieved inodes Extract file data block references Log direct and indirect storage addresses Establish NBD communication channel with host active monitoring phase read request via NBD Locate affected EXT4 structure Register feature accessed in passive monitoring mode write operation issued Determine altered element Document mutation and update internal metadata buffers system partition of 32 GB, and a dedicated 5 GB test disk segment. Prior to each evaluation, a uniform disk snapshot populated with 3,970 files was restored to standardize ex- perimental conditions. Each software instance is executed for a fixed duration of six minutes, during which SHIELD continuously records operational traces from the NBD client, FPGA interface, and underlying file system layer for post- execution examination.

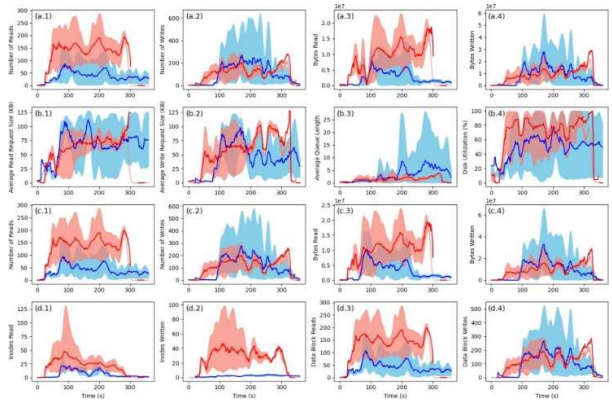

Fig. 3. Temporal visualization of metric variations distinguishing ransomware (red) from benign applications (blue), shown per second. Solid curves repre- sent smoothed averages, while shaded bands indicate metric variability. Panels (a)–(d) correspond to unique and baseline metrics from both NBD and FPGA interfaces.

TABLE I: CATALOG OF MALICIOUS AND NON-MALICIOUS SOFTWARE SAMPLES

| S. No. | Ransomware Variants | Legitimate Applications |
|---|---|---|
| 1 | Intercobros | Veracrypt |
| 2 | AtomSilo | UltraSearch |
| 3 | BlackMatter | Camtasia |
| 4 | Bubuk | GIMP |
| 5 | AvosLocker | 7-Zip |
| 6 | Karma | VLC |
| 7 | Cerber | qBittorrent |
| 8 | Conti | Handbrake |
| 9 | Lockbit | Eraser |
| 10 | Makop | OBS Studio |

*B. Metric Profiles and Comparative Evaluation*

Table II outlines the bifurcation of data features into software-side (via NBD) and hardware-side (collected through the SHIELD interface). While the NBD stream offers reference statistics, hardware-level telemetry enhances granularity and bypasses potential host tampering.

TABLE II: SOFTWARE VS. HARDWARE LAYER METRICS USED FOR DIFFERENTIATION

| Metrics from NBD Channel | Metrics Captured at FPGA Level |
|---|---|
| Read Operation Count | Read Command Volume |
| Write Operation Count | Write Command Volume |
| Data Volume Read | Extracted Read Data |
| Data Volume Written | Captured Write Data |
| Mean Read Length | Quantity of Accessed Inodes |
| Mean Write Length | Total Inodes Modified |
| Average Queue Depth | Count of Data Blocks Accessed |
| Disk Occupancy Ratio | Modified Block Count |

*C. Behavioral Inference from Observed Metrics*

- Analysis from the Software Interface: An examination of metrics collected from the NBD channel suggests a dis- cernible pattern in read-centric activities by ransomware, often surpassing benign processes in both frequency and magnitude. Conversely, the write operations displayed by both categories exhibited marginal differentiation, with overlaps in transac- tional intensity.
- Insights from FPGA-Centric Data Collection: Hardware instrumentation validated software-derived insights and un- veiled additional discriminative cues:
  - Inode Modification Frequency: Ransomware displayed disproportionately high inode alteration rates in contrast to legitimate software, which largely maintained a mini- mal inode footprint.
  - Inode Access Activity: Frequent inode access was in- dicative of ransomware propagation mechanisms, as ob- served consistently across samples.
  - Volume of Data Block Reads: Malicious agents rou- tinely accessed significantly more blocks, typically triple that of standard applications.
  - Write Patterns in Data Blocks: While aggregate write sizes remained comparable, malicious software modified a wider file spectrum.

*D. Enhanced System Realization Pathway*

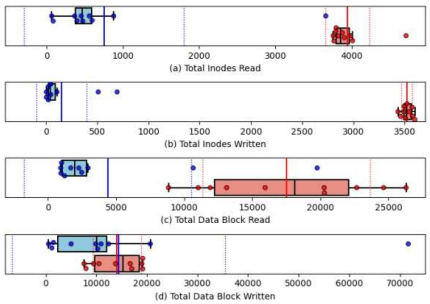


Fig. 4. Comparative statistical trends between malicious (red) and benign (blue) samples using cumulative hardware metrics. Solid and dashed lines denote mean values and standard deviation bands, respectively

While SHIELD demonstrates clear potential in isolating storage-layer anomalies, future implementations should con- sider tighter integration of the FPGA-based monitoring core with kernel-space telemetry extractors. This fusion will facil- itate:

- Real-time decision engines using ML-based anomaly scoring (e.g., random forests, SVMs).
- Embedded logic for file-system journaling events, cur- rently unmonitored in EXT4 structures.
- Isolation of unique ransomware signatures through lon- gitudinal pattern profiling.

Additionally, incorporating on-chip machine learning infer- ence accelerators will reduce the detection latency and allow SHIELD to respond autonomously to evolving ransomware strategies.

### *E. Insights from Correlation and Anomaly Patterns*

- Cross-Layer Metric Consistency: A strong alignment was noted between byte-level metrics and access frequencies across the NBD and FPGA layers. This fidelity reinforces SHIELD's tamper-resilient design and robustness under di- verse operating conditions.
- Identification of Malicious Activity Signatures: Among various parameters, the inode modification rate per unit time stood out as the most reliable indicator of ransomware pres- ence. In contrast with benign utilities—regardless of disk intensity—ransomware systematically altered a majority of available inodes. Patterns of inode reads and data block traver- sals further strengthened classification capability. Although the write activity on blocks exhibited less variance, when paired with inode behavior, it contributed to a reliable fingerprint.

  The statistical distinction between benign and ransomware behavior was validated using metrics such as standard devia- tion, access frequency histograms, and t-tests (p ¡ 0.05). This confirms that inode modification rates and access volume are significant distinguishing factors.

## V. Evaluation and Future Considerations

### *A. Role of Metrics in Ransomware Discrimination*

The experimental framework established that SHIELD en- ables real-time profiling of storage behavior by combining hardware-resident insights with contextual software observa- tions.

Access frequency to low-level EXT4 structures, such as inodes and superblocks, served as reliable indicators of system-level anomalies.

These nuanced signals can be used to implement proactive threat mitigation without reliance on the operating system's integrity.

### *B. Advantages from an Embedded Security Standpoint*

By isolating data acquisition from system software, SHIELD introduces a hardware-based security layer, yielding the following benefits:

- Immunity to Host-Level Manipulation: Since metric capture bypasses the host, evasive or obfuscatory software tactics fail to interfere with SHIELD's operation.
- Local Data Governance: All logs are stored within the enterprise infrastructure, enhancing privacy control and reducing dependency on cloud services.
- Policy-Driven Disk Oversight: Each disk transaction passes through SHIELD, enabling enforcement of real-time access constraints and halting suspect actions on the fly.

### *C. System Limitations and Scalability Pathways*

Although SHIELD proves effective for real-time telemetry, some constraints remain. Tests involving virtualized envi- ronments showed latency peaking at 1 second under active parsing and 0.5 milliseconds during passive observation. De- spite SATA achieving a maximum throughput of 250 MB/s, Ethernet bottlenecks restricted effective bandwidth to roughly 100 KB/s.

Performance enhancements may be achieved via:

- Packet Transmission Enhancement: Elevating the de- fault Ethernet packet size to 1500 bytes—or utilizing a custom transmission protocol—could alleviate throughput limitations.
- Buffer Capacity Scaling: Increasing the FPGA's internal buffer sizes from the current 4–16 KB range to 32–64 KB can reduce context-switching delays and optimize transaction throughput.
- Advanced FPGA Migration: Transitioning to newer platforms such as the Xilinx 7 Series could unlock higher clock rates, expanded memory, and high-speed peripheral compatibility (USB 3.0 or PCIe), facilitating better scalability.

## VI. Related Work

Contemporary threat detection methodologies encompass diverse operational layers within computing infrastructures and have progressed significantly from early signature-centric defenses. Traditional mechanisms, such as file authenticity ver- ification tools and classical antivirus engines, are designed to recognize static indicators of compromise or unauthorized al- terations. However, these approaches often suffer from latency in detection and are prone to circumvention by polymorphic or zero-day malware variants.

Advanced frameworks incorporating data-driven paradigms—especially those grounded in machine learning or pattern recognition—demonstrate the capacity to uncover previously unseen attack vectors. Nonetheless, such systems predominantly depend on telemetry obtained from within the host environment, exposing them to the risk of subversion or manipulation by resident malware.

Solutions deployed at the operating system kernel tier permit visibility into fundamental runtime operations and low- level events. Despite their deeper introspection capabilities, these systems can negatively affect system responsiveness and remain susceptible to breaches affecting privileged software layers. Composite solutions that combine insights from both user-space applications and kernel subsystems enhance be- havioral context but do not provide isolation from potentially compromised host environments.

Architectures utilizing hardware-facilitated indicators, such as processor execution traces and microarchitectural perfor- mance metrics, offer resistance against software tampering. While effective for safeguarding control flow integrity and detecting anomalies in execution patterns, these techniques frequently lack awareness of higher-layer abstractions such as file system modifications and disk structure evolution.

Externally positioned analytical tools—ranging from virtu- alized sandboxing frameworks to cloud-based forensic utili- ties—support retrospective evaluation and enterprise-wide in- cident correlation. However, they often fall short in ensuring data privacy compliance and cannot offer continuous, fine- grained inspection of storage subsystem behavior. Addition- ally, their effectiveness is constrained when rapid response or localized observability is essential.

To address the aforementioned deficiencies, the architecture introduced in this study employs a dedicated hardware-based monitoring module capable of tracking file system-level activi- ties independently of the host's operational state. By intercept- ing and analyzing storage operations at the physical interface layer, this solution ensures resilient, low-latency telemetry collection that remains impervious to host-side interference. In doing so, it introduces a paradigm shift toward real-time, autonomous, and tamper-proof threat detection grounded in storage behavior profiling.

## VII. Conclusion and Future Work

The architecture introduced under the SHIELD framework demonstrates a resilient and extensible approach to ran- somware identification through the extraction of granular file system and device-level behavioral indicators. By decoupling the monitoring mechanism from the host environment and leveraging a customized SATA Host Bus Adapter in conjunc- tion with a Network Block Device (NBD) protocol, the system enables detailed introspection of storage interactions with high fidelity and without host dependency.

This method facilitates robust discrimination between legiti- mate applications and malicious executables by analyzing sub- tle patterns within storage-level operations. The architectural independence ensures immunity from host-side manipulation, thereby reinforcing the reliability of the collected metrics. The ability to detect aberrant access trends across inode and data block structures highlights SHIELD's effectiveness in identifying early signs of ransomware behavior before payload execution completes.

Planned future iterations of this platform involve the in- corporation of adaptive learning algorithms capable of autonomously categorizing behaviors based on evolving threat signatures. Further, embedding the monitoring logic directly within silicon-based storage controller components is envi- sioned, which would support continuous surveillance at hard- ware speeds while minimizing system latency. Such integration would empower storage solutions to act as proactive security endpoints, contributing to real-time mitigation of advanced persistent threats and offering a fortified defense posture in high-assurance computing environments.

## References


[1] A. Young and M. Yung, "Cryptovirology: extortion-based security threats and countermeasures," in IEEE Symposium on Security and Privacy, 1996, pp. 129–140.

[2] P. H. Meland, Y. F. F. Bayoumy, and G. Sindre, "The Ransomware-as- a-Service economy within the darknet," Computers & Security, vol. 92, p. 101762, 2020.

[3] Cybots AI, “The Road to Ransomware Resilience: Behaviour Analysis,” 2024. [Online]. Available: https://cybotsai.com/the-road-to-ransomware- resilience-behaviour-analysis/
[4] K. Townsend, “Ransomware in 2024: More Attacks, More Leaks, and Increased Sophistication,” 2024. [Online]. Available: https://www.securityweek.com/ransomware-in-2024-more-attacks-more- leaks-and-increased-sophistication/
[5] IBM, “What Is Ransomware-as-a-Service (RaaS)?” 2024. [Online]. Available: https://www.ibm.com/topics/ransomware-as-a-service
[6] J. Lyons, “Lockbit wins ransomware speed test, encrypts 25,000 files per minute,” 2022. [Online]. Available: https://tinyurl.com/3hccp4fm
[7] M. A. Putrevu et al., “A Comprehensive Analysis of Machine Learning Based File Trap Selection Methods to Detect Crypto Ransomware,” arXiv preprint arXiv:2409.11428, 2024.
[8] Splunk, “A Comparative Analysis of Ransomware Encryption Speed,” 2022. [Online]. Available: https://tinyurl.com/ymnk248v
[9] A. Kharaz et al., “UNVEIL: A Large-Scale, Automated Approach to Detecting Ransomware,” in USENIX Security Symposium, 2016, pp. 757–772.
[10] Z.-G. Chen et al., “Automatic Ransomware Detection and Analysis Based on Dynamic API Calls Flow Graph,” in Int. Conf. on Research in Adaptive and Convergent Systems (RACS), 2017, pp. 196–201.
[11] P. M. Anand, P. S. Charan, and S. K. Shukla, “HiPeR-Early Detection of a Ransomware Attack Using Hardware Performance Counters,” Digital Threats: Research and Practice, vol. 4, no. 3, pp. 1–24, 2023.
[12] G. O. Ganfure et al., “Rtrap: Trapping and Containing Ransomware with Machine Learning,” IEEE Trans. on Information Forensics and Security, vol. 18, pp. 1433–1448, 2023.
[13] J. A. Gomez-Hernandez et al., “Inhibiting Crypto-Ransomware on Win- dows Platforms Through a Honeyfile-Based Approach with R-Locker,” IET Information Security, vol. 16, no. 1, pp. 64–74, 2022.
[14] V. Kotov and M. Rajpal, “Understanding Crypto-Ransomware,” 2023. [Online]. Available: https://arxiv.org/abs/2312.07641
[15] C. Beaman et al., “Ransomware: Recent advances, analysis, challenges and future research directions,” Comput. Secur., vol. 111, Dec. 2021.
[16] P. M. Anand et al., “RTRShield: Early Detection of Ransomware Using Registry and Trap Files,” in Information Security Practice and Experience, Springer Nature, 2023, pp. 209–229.
[17] M. Sohail and S. Tabet, “Data privacy and ransomware impact on cyber-physical systems data protection,” in Cyber-Physical Systems for Industrial Transformation, CRC Press, 2023, pp. 115–134.
[18] J. Ispahany et al., “Ransomware Detection Using Machine Learning: A Review, Research Limitations and Future Directions,” IEEE Access, vol. 12, pp. 68785–68813, 2024.
[19] A. Mathur et al., “The new ext4 filesystem: Current status and future plans,” in Linux Symposium, 2007.
[20] P. Breuer, A. Mar´ın-Lo´pez, and A. Ares, “The Network Block Device,” Linux Journal, vol. 73, 2000.
[21] B. Djordjevic and V. Timcenko, “Ext4 file system performance analysis in linux environment,” in WSEAS Int. Conf. on Applied Informatics and Communications, 2011, pp. 288–293.
[22] S. Dara, “Understanding Ext4 Disk Layout, Part 1,” 2023. [On- line]. Available: https://blogs.oracle.com/linux/post/understanding-ext4- disk-layout-part-1
[23] Missing Link Electronics, “SATA Storage Extension for ZYNQ 7000,” 2024. [Online]. Available: https://www.missinglinkelectronics.com/ip- cores/sata-storage-extension-for-zynq-7000/
[24] K. Eguro, “SIRC: An Extensible Reconfigurable Computing Communi- cation API,” in IEEE FCCM, 2010, pp. 135–138.
[25] L. Woods and K. Eguro, “Groundhog - A Serial ATA Host Bus Adapter (HBA) for FPGAs,” in IEEE FCCM, 2012, pp. 220–223.
[26] Fpgasystems, “Groundhog: Serial ATA Host Bus Adapter,” 2013. [On- line]. Available: https://github.com/fpgasystems/groundhog
[27] A. Cozzette, “BUSE: A Block Device in User Space for Linux,” 2013. [Online]. Available: https://github.com/acozzette/BUSE
[28] Serial ATA International Organization, Serial ATA Revision 3.1 Gold Revision, 2011. [Online]. Available: https://sata- io.org/system/files/specifications/SerialATA
[29] AMD, ML505/ML506/ML507 Evaluation Platform User Guide (UG347) 3.1.2, May 2011. [Online]. Available: https://docs.amd.com/v/u/en- US/ug347
[30] Digilent, “Virtex-5 OpenSPARC,” 2024. [Online]. Available: https://digilent.com/reference/programmable-logic/virtex-5/start
[31] NetworkBlockDevice, “Network Block Device,” 2024. [Online]. Avail- able: https://github.com/NetworkBlockDevice/nbd
[32] N. Murphy, “What is File Integrity Monitoring (FIM) and Why Is It Im- portant?” 2024. [Online]. Available: https://www.lepide.com/blog/what- is-file-integrity-monitoring/
[33] V. S. Sathyanarayan, P. Kohli, and B. Bruhadeshwar, “Signature Gener- ation and Detection of Malware Families,” in Information Security and Privacy, Springer, 2008, pp. 336–349.
[34] H. Zhang et al., “Ranker: Early Ransomware Detection Through Kernel- Level Behavioral Analysis,” IEEE Trans. on Information Forensics and Security, vol. 19, pp. 6113–6127, 2024.

[35] R. Elnaggar et al., “Runtime Malware Detection Using Embedded Trace Buffers,” IEEE Trans. on CAD of Integrated Circuits and Systems, vol. 41, no. 1, pp. 35–48, 2022.
[36] C. Malone, M. Zahran, and R. Karri, “Are hardware performance counters a cost effective way for integrity checking of programs,” in ACM Workshop on Scalable Trusted Computing, 2011, pp. 71–76.
[37] B. Denham and D. R. Thompson, “Ransomware and Malware Sandbox- ing,” in IEEE UEMCON, 2022, pp. 0173–0179.
[38] Microsoft, “AI-Driven Adaptive Protection Against Human-Operated Ransomware,” 2021. [Online]. Available: https://www.microsoft.com/en- us/security/blog/2021/11/15/ai-driven-adaptive-protection-against- human-operated-ransomware/
[39] B. Reidys et al., “RSSD: defend against ransomware with hardware- isolated network-storage codesign and post-attack analysis,” in ACM ASPLOS, 2022, pp. 726–739.
[40] A. Continella et al., “ShieldFS: a self-healing, ransomware-aware filesystem,” in Annual Conf. on Computer Security Applications, ACM, 2016, pp. 336–347.
[41] D. Dimov and Y. Tsonev, “Observing, Measuring and Collecting HDD Performance Metrics on a Physical Machine During Ransomware At- tack,” Information & Security: An International Journal, vol. 47, no. 3, pp. 317–327, 2020.
[42] CrowdStrike, “Malware Analysis: Steps & Examples,” 2023. [Online]. Available: https://www.crowdstrike.com/en-us/cybersecurity- 101/malware/malware-analysis/
[43] M. A. Al-Garadi et al., “A Survey of Machine and Deep Learning Methods for Internet of Things (IoT) Security,” IEEE Commun. Surveys & Tutorials, vol. 22, no. 3, pp. 1646–1685, 2020.
[44] Z. Pan et al., “Hardware-Assisted Malware Detection using Machine Learning,” in DATE, 2021, pp. 1775–1780.
[45] S. Razaulla et al., “The Age of Ransomware: A Survey on the Evolu- tion, Taxonomy, and Research Directions,” IEEE Access, vol. 11, pp. 40698–40723, 2023.
[46] J. Singh, “IHPP: An In-Depth Profiling Tool for Advanced Perfor- mance Analysis,” PMJ, vol. 35, no. 1, 2024. [Online]. Available: https://doi.org/10.52783/pmj.v35.i1.5051
[47] J. Singh, “Web Application Development and Cybersecurity: Integrating APIs, Logging, and Defense Mechanisms,” PMJ, vol. 35, no. 1, 2024. [Online]. Available: https://doi.org/10.52783/pmj.v35.i1.5049
[48] . Singh, “Transactional Memory without Hardware Support: A Compre- hensive Software-Driven Approach,” PMJ, vol. 35, no. 1, 2024. [Online]. Available: https://doi.org/10.52783/pmj.v35.i1.5048
[49] J. Singh, “Comparative Analysis of Open-Source and Closed-Source Development Models Using Agent-Based Simulation,” PMJ, vol. 35, no. 1, 2024. [Online]. Available: https://doi.org/10.52783/pmj.v35.i1.5047
[50] R. U. B. Rizan, “Evaluating Object Tracking Algorithms,” in Proc. 2024 Int. Conf. on Engineering and Emerging Technologies (ICEET), pp. 1–7, 2024. doi: 10.1109/ICEET65156.2024.10913572
[51] R. U. B. Rizan, “Enhancing Edge Detection in Images Using Ant Colony Optimization,” in Proc. 21st Int. Conf. on Computing and Information Technology (IC2IT 2025), Springer Nature Switzerland, Cham, pp. 182–192, 2025. isbn: 978-3-031-90295-6
[52] R. U. B. Rizan, “Advancing Image Segmentation and Classification with Mamba-Based Architectures,” in Proc. 21st Int. Conf. on Computing and Information Technology (IC2IT 2025), Springer Nature Switzerland, Cham, pp. 134–144, 2025. isbn: 978-3-031-90295-6
[53] R. Recharla, “Building a Scalable Decentralized File Exchange Hub Using Google Cloud Platform and MongoDB Atlas,” in Proc. 2025 Int. Conf. on Wireless Communications Signal Processing and Networking (WiSPNET), pp. 1–7, 2025. doi: 10.1109/WiSP- NET64060.2025.11005333
[54] R. Recharla, “FlexAlloc: Dynamic Memory Partitioning for SeKVM,” in Proc. 2025 Int. Conf. on Wireless Communications Signal Process- ing and Networking (WiSPNET), pp. 1–9, 2025. doi: 10.1109/WiSP-NET64060.2025.11004912
[55] R. Recharla, “Parallel Sparse Matrix Algorithms in OCaml v5: Imple- mentation, Performance, and Case Studies,” in Proc. 2025 Int. Conf. on Wireless Communications Signal Processing and Networking (WiSP- NET), pp. 1–9, 2025. doi: 10.1109/WiSPNET64060.2025.11004864
[56] Unknown Author, “Machine Learning (cs.LG); Artificial Intelli- gence (cs.AI),” arXiv:2502.04418 [cs.LG], 2025. [Online]. Available: https://doi.org/10.48550/arXiv.2502.04418