International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 06 , June - 2026

A Comprehensive Survey of Machine Learning - Based Ransomware Detection: Taxonomy, Behavioral Analysis, and Emerging Defense Paradigms

Mahmoud Al-Rahhal
Department of Computer Systems and Networks
Faculty of Informatics Engineering
University of Aleppo, Syria

Souheil Khawatmi
Department of Computer Systems and Networks
Faculty of Informatics Engineering
University of Aleppo, Syria

Fadi Farha
Department of Computer Systems and Networks
Faculty of Informatics Engineering
University of Aleppo, Syria

Mohamed M. Khatib
Department of Artificial Intelligence
Faculty of Informatics Engineering
University of Aleppo, Syria

Abstract - Ransomware has emerged as one of the most destructive and rapidly evolving cyber threats, causing financial loss, operational disruption, and risks to critical infrastructure. Modern ransomware increasingly employs obfuscation, sandbox evasion, adaptive execution, and multi-stage attack strategies, limiting the effectiveness of signature-based defenses. Consequently, machine learning (ML) and artificial intelligence (AI) have become central to behavior-driven ransomware detection, supporting early-stage detection and improved generalization to unseen variants. This paper presents a comprehensive and systematic survey of ML-based ransomware detection techniques. It introduces a unified taxonomy that classifies existing approaches according to four dimensions: target platform, analysis level, learning paradigm, and detection stage. Guided by this taxonomy, the survey critically reviews static analysis, dynamic behavior modeling, hybrid static–dynamic frameworks, and platform-specific techniques across Windows, Android, Linux, IoT, cloud, and cross-platform environments. It also examines advanced paradigms, including deep representation learning, hardware-assisted detection, graph-based modeling, explainable AI, concept drift–aware learning, and pre-encryption detection. Beyond algorithms, the study analyzes datasets, feature engineering practices, and evaluation methodologies, highlighting persistent challenges related to reproducibility, dataset obsolescence, adversarial evasion, and deployment feasibility. Following the PRISMA 2020 protocol and five research questions, this survey synthesizes more than a decade of research to identify key trends, open challenges, and future directions for robust, explainable, and deployable real-world ransomware defense systems.

Keywords - Ransomware Detection; Machine Learning; Dynamic Analysis; Static Analysis; Pre-Encryption Detection; Concept Drift; Explainable AI; Hardware-Assisted Security; Graph-Based Learning; Cybersecurity

1. INTRODUCTION

Ransomware has become one of the most disruptive and financially damaging cyber threats of the past decade, affecting individuals, enterprises, public-sector organizations, and critical infrastructure worldwide. Unlike conventional malware, it is explicitly monetized through data encryption, operational disruption, and extortion, increasingly augmented by data exfiltration and double or triple extortion [1]. Driven by professionalized criminal ecosystems and ransomware-as-a-service (RaaS) models, ransomware has shifted from an isolated technical nuisance into an industrialized extortion economy, eroding the value of signature-based detection and reactive defense [1], [2]. In response, research has shifted toward behavior- and learning-based detection [1]. Machine learning (ML) and artificial intelligence (AI) now underpin many ransomware detectors, enabling automated feature extraction, early-stage behavioral analysis, and improved generalization to previously unseen variants [3], [4]. However, the rapid expansion of this domain—across platforms, behavioral signals, learning paradigms, datasets, and evaluation practices—has produced a fragmented literature in which incompatible datasets, metrics, and threat models make reported gains difficult to compare or reproduce, a concern revisited in Sections 7 and 8. This survey addresses that fragmentation through a structured, critical synthesis of machine learning–based ransomware detection.

1.1 Motivation and Impact of Ransomware

Ransomware's consequences extend beyond financial loss to business continuity, data confidentiality, public trust, and—in sectors such as healthcare, energy, and transportation—human safety. Contemporary campaigns emphasize pre-encryption reconnaissance, lateral movement, credential abuse, and stealthy persistence; detection that activates only after encryption begins is therefore often too late. This temporal asymmetry shifts the objective from post-hoc classification to time-critical, pre-encryption prevention [2], [5], [6]. Ransomware also resists conventional defenses. Many benign applications perform cryptographic operations, bulk file modification, and privilege escalation, which undermines rule-based heuristics [7], while adversaries evade sandboxes, delay execution, inject benign-looking noise, and continuously drift their tactics, techniques, and procedures (TTPs) [8], [9]. Effective detectors must therefore be behavior-aware, adaptive, explainable, and evasion-resilient [10], [11] a combination that few individual systems achieve simultaneously, as subsequent sections show.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 06 , June - 2026

1.2 Evolution and Taxonomy of Ransomware and Defenses

Ransomware has evolved from early locker-style malware into cryptographic extortion platforms employing hybrid encryption, anonymized payment infrastructures, automated propagation, and multi-stage lifecycles spanning initial access, reconnaissance, privilege escalation, lateral movement, exfiltration, encryption, and extortion. Defenses have correspondingly progressed from signature-based antivirus toward behavior-centric, multi-layer architectures [12], [13]. Existing taxonomies organize ransomware by delivery vector, encryption strategy, target platform, propagation, and extortion model, and detection by static, dynamic, network-centric, memory- and hardware-level, and hybrid analysis. Although recent work increasingly emphasizes cross-layer correlation, early detection under partial observability, and adversarial robustness, inconsistent terminology, evaluation protocols, and dataset usage still impede fair comparison. Prevailing schemes also remain predominantly structural; the behavior- and stage-centric taxonomy introduced in Section 2 instead foregrounds the early-detection and concept-drift dimensions that earlier schemes underweight [1], [2].

1.3 Role of Machine Learning and AI in Ransomware Detection

Machine learning and AI are central to ransomware detection because they model complex behavioral patterns, generalize beyond handcrafted signatures, and adapt to evolving threats. Classical techniques—Random Forests, Support Vector Machines, and gradient boosting—perform strongly on engineered features from API and system calls, file-system activity, and network traffic [3], [14], [15], whereas deep models (CNNs, RNNs, Transformers, and hybrids) enable representation learning over sequential, structural, and multimodal behavioral data [16], [17], [18], [19]. Beyond accuracy, recent work targets explainability, concept drift, streaming analysis, and pre-encryption latency [6], [9], [20]: explainable AI supports operational trust, forensic analysis, and compliance [21], [22]; drift-aware learning sustains performance under evolving behavior; and hardware- and memory-centric methods extend visibility beyond software-level observables [23], [24], [25]. Persistent gaps in dataset availability, reproducibility, evaluation consistency, adversarial robustness, and deployment feasibility indicate that the frontier has shifted from maximizing accuracy to sustaining it under drift, adversarial pressure, and tight latency budgets [1], [2].

1.4 Scope of This Review and Paper Selection Methodology

This review provides a systematic and in-depth analysis of machine learning–based ransomware detection research with the following objectives:

- Comprehensive coverage of static, dynamic, hybrid, and advanced ransomware detection techniques across Windows, Android, Linux, IoT, cloud, and cross-platform environments.
- Behavior-centric organization of the literature, emphasizing feature sources, learning paradigms, and detection objectives such as early-stage and pre-encryption detection.
- Critical examination of datasets, feature engineering practices, evaluation methodologies, and robustness under concept drift and adversarial conditions.
- Identification of emerging trends and open challenges, including explainable detection, hardware-assisted monitoring, graph-based modeling, and zero-shot or generalization-driven approaches.

Research questions. To structure this synthesis and to make its scope auditable, the review is guided by five research questions (RQs):

RQ1. Which feature sources, analysis levels, and learning paradigms dominate ML-based ransomware detection, and how do they map onto target platforms?

RQ2. To what extent do current approaches enable early-stage and pre-encryption detection rather than post-encryption classification?

RQ3. How robust are existing detectors to concept drift and adversarial evasion, and how rigorously is this robustness evaluated?

RQ4. Which datasets, feature-engineering practices, and evaluation protocols underpin reported results, and how do they affect reproducibility and real-world deployability?

RQ5. What are the principal open challenges and the most promising directions for deployable, explainable ransomware defense?

Eligibility criteria. Study eligibility was governed by explicit criteria. A study was included if it (i) proposed or evaluated a machine- or deep-learning ransomware detection or pre-encryption technique, (ii) reported a reproducible methodology with a quantitative evaluation, and (iii) appeared in a peer-reviewed venue published between 2017 and 2026. A study was excluded if it (i) addressed generic malware or botnets without ransomware-specific analysis, (ii) contributed no ML or DL detection component (for example, a purely forensic or recovery focus), (iii) duplicated or was clearly superseded by stronger included work, or (iv) provided insufficient methodological detail for reliable assessment.

Search and selection. The structured search was executed in Scopus—which indexes the major digital libraries relevant to this domain (IEEE Xplore, the ACM Digital Library, ScienceDirect, SpringerLink, and MDPI)—using queries combining terms such as ransomware detection, static and dynamic analysis, machine learning, deep learning, pre-encryption detection, concept drift, and behavioral analysis, and was complemented by backward and forward citation tracking to recover seminal works not surfaced by keyword search. As summarized in Figure 1, the initial search returned 146 records; 55 were removed during screening, yielding the 91 studies synthesized in this review. Approximately 54% of these were published between 2023 and 2026 and only about 12% before 2020, underscoring both the field's rapid maturation and the attendant risk of dataset and benchmark obsolescence examined in Section 7.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 06 , June - 2026

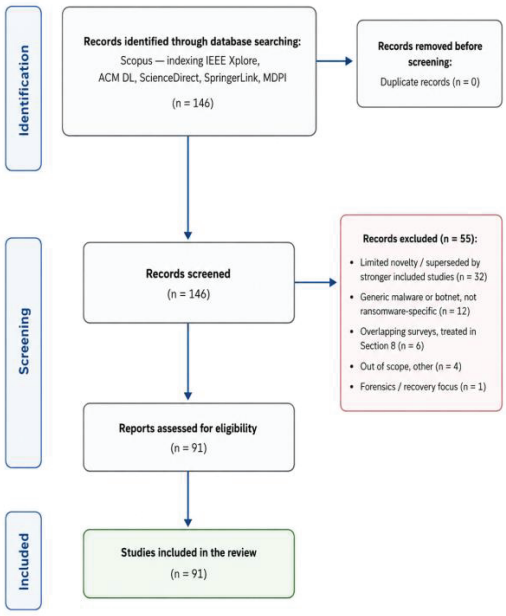

Fig. 1. PRISMA 2020 flow diagram of the study-selection process.

The remainder of this review is organized as follows. Section 2 introduces the taxonomy that organizes the survey. Sections 3 and 4 examine static and dynamic ransomware detection techniques, respectively, followed by platform-specific analyses in Section 5. Section 6 reviews advanced and emerging detection paradigms, including deep learning, hardware-assisted monitoring, graph-based modeling, and adversarial robustness. Section 7 discusses datasets, feature engineering practices, and evaluation methodologies, while Section 8 synthesizes insights from existing surveys and outlines future research directions.

2 TAXONOMY OF MACHINE LEARNING–BASED RANSOMWARE DETECTION

The diversity of ransomware behaviors, execution environments, and detection objectives has resulted in a broad spectrum of machine learning (ML)–based ransomware detection approaches. These approaches differ significantly in terms of feature sources, analysis depth, learning models, and operational assumptions. To enable systematic comparison and to provide a unifying framework for the detailed review presented in Sections 3–8, this section introduces a taxonomy of ML-based ransomware detection along four key dimensions: target platform, analysis level, learning paradigm, and detection stage. These four dimensions are consolidated in the unified taxonomy presented in Figure 2, which serves as the organizing framework for the detailed technical review that follows.

2.1 Dimensions of Classification

2.1.1 *Target Platform*

The target platform strongly influences both ransomware behavior and the availability of observable features. Existing ML-based ransomware detection approaches can be categorized according to the platform they are designed to protect:

- Windows-based systems, which provide rich telemetry such as Portable Executable (PE) file structures, Windows Application Programming Interface (API) calls, registry operations, Event Tracing for Windows (ETW) logs, and file-system events.
- Android and mobile platforms, characterized by permission-based security models, application sandboxing, and constrained system visibility.
- Linux platforms, increasingly targeted in server, cloud, and containerized environments, often requiring platform-agnostic and low-level behavioral signals.
- Internet of Things (IoT) and Cyber–Physical Systems (CPS), where resource constraints, device heterogeneity, and safety-critical operation necessitate lightweight and memory- or network-centric detection mechanisms.
- Cross-platform frameworks, which aim to generalize detection across operating systems by focusing on behavioral semantics rather than operating-system–specific artifacts.
- Network-only detection approaches, which infer ransomware activity exclusively from network traffic patterns, encrypted communication metadata, or propagation behavior.

This classification highlights the trade-off between detection fidelity and deployability, and explains why techniques effective on desktop platforms may not be directly applicable to mobile, cloud, or IoT environments.

2.1.2 Analysis Level

The analysis level defines where behavioral evidence is extracted and reflects the depth of system visibility available to the detector:

- Static analysis, examining binaries, bytecode, opcodes, permissions, and structural metadata without executing the program.
- Dynamic analysis, monitoring runtime behavior such as API calls, system calls, file-system activity, registry modifications, and memory usage.
- Hybrid static–dynamic analysis, combining pre-execution artifacts with runtime behavior to improve robustness against obfuscation and evasion.
- Memory-level analysis, focusing on volatile artifacts such as decrypted code segments, runtime buffers, injected payloads, and in-memory execution traces.
- Hardware-level analysis, leveraging performance monitoring counters and microarchitectural signals that are largely independent of the operating system.
- Network-level analysis, capturing command-and-control communication, encrypted traffic characteristics, and lateral movement behavior.
- Storage and file-system–level analysis, modeling disk access patterns, encryption behavior, write operations, and entropy evolution.

This dimension reflects a progression from lightweight and early-stage detection toward more evasive-resistant but operationally complex monitoring strategies.

2.1.3 Learning Paradigm

ML-based ransomware detection has evolved through multiple learning paradigms, each addressing different operational requirements and threat characteristics:

- Classical machine learning, including Random Forest, Support Vector Machines (SVMs), k-nearest neighbors (k-NN), and gradient boosting, valued for computational efficiency and interpretability.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 06 , June - 2026

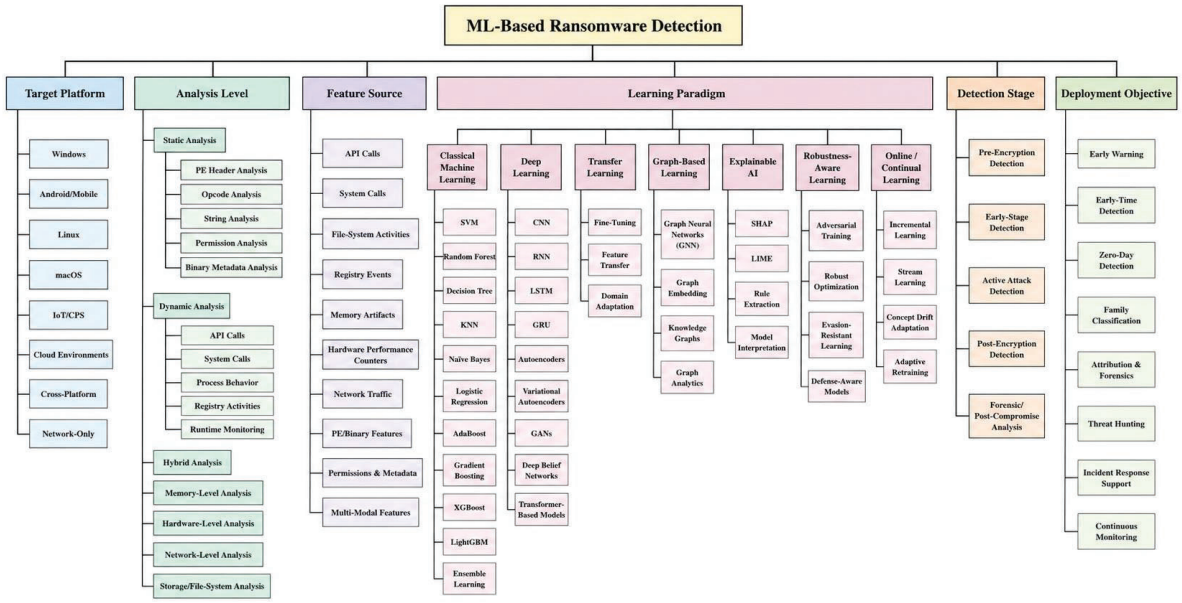

Fig. 2. Unified taxonomy of ML-based ransomware detection along four dimensions.

- Deep learning, such as Convolutional Neural Networks (CNNs), Long Short-Term Memory (LSTM) networks, and Transformer-based models, enabling automated feature extraction from sequential, high-dimensional, and multimodal data.
- Transfer learning and representation learning, leveraging pretrained models or semantic embeddings to improve generalization across ransomware families and platforms.
- Explainable artificial intelligence (XAI), incorporating methods such as SHapley Additive exPlanations (SHAP) and Local Interpretable Model-agnostic Explanations (LIME) to enhance transparency, trust, and forensic interpretability.
- Graph-based learning, using graph neural networks and contrastive learning to capture structural dependencies and relational behavior.
- Blockchain-enabled frameworks, focusing on trust, auditability, data integrity, and decentralized coordination in distributed detection environments.
- Adversarial- and robustness-aware learning, explicitly addressing evasion strategies, noise injection, cooperative attacks, and adaptive ransomware behavior.

This dimension reflects a shift from accuracy-centric detection toward robustness, interpretability, and long-term deployability.

2.1.4 Detection Stage

The detection stage characterizes when a ransomware detector operates relative to the ransomware attack lifecycle:

- Pre-encryption detection, aiming to prevent irreversible encryption and data loss.
- Early-stage detection, operating under partial observability and short execution windows.
- Post-encryption detection, primarily supporting containment, alerting, and response.
- Forensic and post-compromise analysis, focusing on attribution, behavioral reconstruction, and incident investigation.

Recent research increasingly prioritizes pre-encryption and early-stage detection, as post-encryption identification alone is insufficient to prevent operational and financial damage.

2.2 Overall Trends and Research Gaps

Analysis of the literature through the proposed taxonomy reveals several overarching trends. First, there is a clear transition from static and signature-inspired approaches toward behavior-driven and multi-layer detection frameworks, driven by widespread obfuscation, packing, and sandbox evasion. Second, dynamic sequence modeling over API calls and system calls has become dominant, although such approaches remain vulnerable to delayed execution, partial observability, and cooperative multi-process attacks. Third, increasing attention is given to low-level behavioral signals, including memory artifacts, storage access patterns, and hardware performance counters, which offer strong evasion resistance but introduce deployment and scalability challenges.

Despite these advances, significant research gaps remain. Windows-centric detection continues to dominate the literature, while Linux-, IoT-, and cross-platform ransomware detection remain comparatively underexplored. Few systems simultaneously address early detection, concept drift adaptation, and explainability, limiting long-term operational viability. Moreover, evaluation practices remain inconsistent, with limited emphasis on pre-encryption latency, drift-aware testing, and adversarial robustness.

These trends and gaps motivate the detailed technical analysis presented in Sections 3–8, where the taxonomy introduced here is instantiated across static, dynamic, hybrid, platform-specific, and emerging ransomware detection paradigms.

3 STATIC ANALYSIS–BASED RANSOMWARE DETECTION

Static analysis detects ransomware from the structural and semantic properties of a binary without executing it, on the premise that obfuscation, packing, and polymorphism cannot

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 06 , June - 2026

fully erase the invariants imposed by the attack objective—file enumeration, encryption preparation, key management, persistence, and command-and-control (C2) communication. These invariants surface in static artifacts such as opcode distributions, imported functions, permission declarations, control-flow structures, and binary metadata, which expose indirect evidence of malicious intent for crypto-ransomware, locker variants, and hybrid droppers even when execution is delayed or environment-aware. This makes static screening a natural first line of defense in pre-execution and pre-deployment contexts, where runtime monitoring is impractical or would act too late to prevent encryption.

Static features span a spectrum of abstraction: low-level byte and opcode representations capture fine-grained semantics but are sensitive to syntactic transformation, whereas structural features (APIs, permissions, and dependency or control-flow graphs) offer greater interpretability and resilience at the cost of behavioral context. The principal advantage of static analysis is that it sidesteps sandbox evasion, delayed payload activation, environment fingerprinting, and path explosion, enabling pre-execution screening at scale; its principal blind spot is runtime-dependent behavior—conditional encryption triggers, network-driven key retrieval, and adaptive execution—that remains invisible to static inspection.

Within detection pipelines, static analysis therefore plays three roles: pre-execution detection that reduces reliance on post-compromise response, platform-scalable screening for enterprise, mobile, and IoT settings, and a complementary prior that guides runtime monitoring and feature prioritization in hybrid systems. The following subsections review opcode-, bytecode-, and binary-level representations, structural and API-based features, and the foundational studies that established static analysis as a core component of ransomware defense.

3.1 Opcode, Bytecode, and Binary-Level Feature Analysis

Opcode- and bytecode-based representations remain among the most widely explored static features for ransomware detection due to their ability to capture instruction-level semantics while remaining resilient to superficial code obfuscation and packing.

Alonso et al. [26] investigate Linux malware detection by comparing static binary inspection against runtime opcode tracing. While the dynamic Random Forest classifier achieves over 90% accuracy, the static Long Short-Term Memory (LSTM) model trained on opcode sequences reaches 98% accuracy, demonstrating that well-modeled static opcode sequences can rival dynamic approaches. Although the study is not ransomware-specific, it is highly relevant due to its focus on non-Windows platforms, which are increasingly targeted by modern ransomware campaigns.

Recent work has explored representation learning over static artifacts. Kesharwani et al. [16] propose a behavior-to-vision transformation in which static behavioral reports are encoded as red–green–blue (RGB) images and classified using Vision Transformers (ViTs). The ViT-based model achieves 99.78% accuracy, outperforming convolutional neural network (CNN) baselines, indicating that static visual encodings can effectively capture complex ransomware characteristics even in evolving Android ecosystems.

Several studies focus explicitly on opcode *n*-gram modeling. Zhang et al. [27] demonstrate that static *n*-gram opcode features combined with term frequency–inverse document frequency (TF–IDF) weighting can achieve 99.3% binary classification accuracy and strong multi-class performance, particularly for WannaCry variants. Extending this line of work, Zhang et al. [28] introduce a patch-based CNN with self-attention mechanisms to process long opcode sequences efficiently. Their approach outperforms conventional CNN and recurrent neural network (RNN) architectures and demonstrates robustness against sandbox-aware ransomware.

Ensemble learning has also been shown to strengthen static detection performance. Johnson et al. [29] propose an ensemble framework based on opcode *n*-grams and TF–IDF feature selection, outperforming individual classifiers such as Support Vector Machines (SVMs) and Random Forests. These results confirm that static ensemble models remain valuable complements to dynamic analysis in hybrid ransomware defense systems.

Beyond classification, static analysis has been applied to ransomware clustering and attribution. Yamany et al. [30] propose a static indexing and clustering scheme using hybrid binary features, demonstrating that Import Address Table (IAT) features outperform string-based features for ransomware family discrimination. In a later extension, Yamany et al. [31] integrate static, dynamic, and visualization-based similarity analysis, showing that multi-perspective classification improves robustness against packing and polymorphism.

Static feature optimization has also been explored in resource-constrained environments. Alissa et al. [32] combine Dwarf Mongoose Optimization with Extreme Learning Machines for Internet of Things (IoT) ransomware detection, reducing feature dimensionality while maintaining high detection accuracy.

3.2 Static API, Permission, and Structural Feature Analysis

Static analysis of API usage, permissions, and structural dependencies provides interpretable indicators of ransomware intent, particularly in Windows and Android environments.

Mohan Anand et al. [14] conduct a large-scale static API importance analysis across 46 ransomware families, identifying 135 discriminative Windows API calls. Their results show that API-level static features enable lightweight and interpretable ransomware detection with 96.15% accuracy, making them suitable for deployment in endpoint security systems.

Zirari et al. [33] present a comprehensive behavioral dataset derived from ransomware execution, emphasizing strategic API usage patterns. Although their analysis includes dynamic artifacts, the extracted API statistics and structural patterns can be leveraged in static-informed detection pipelines, serving as a valuable resource for future machine learning (ML) and deep learning (DL) models.

Graph-based static representations have recently gained attention. Satpathy and Swain [17] propose Graph-Contrast Ransomware Detection (GCRD), modeling feature dependencies within Portable Executable (PE) files using graph neural networks and contrastive learning. Their Transformer-based classifier achieves 99.1% accuracy with low inference latency, highlighting the promise of graph-based static modeling for early-stage ransomware detection.

On mobile platforms, Gharib and Ghorbani [34] introduce DNA-Droid, a hybrid Android ransomware detection framework in which static permissions and application metadata guide deeper behavioral analysis. The framework achieves high precision and recall, demonstrating that static permission analysis remains effective as a first-line defense on Android systems.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 06 , June - 2026

TABLE I. : STATIC RANSOMWARE ANALYSIS: FEATURE ABSTRACTIONS, THEORETICAL PROPERTIES, AND REPRESENTATIVE STUDIES

Static Analysis Category	Feature Representation	Theoretical Strengths	Key Limitations	Representative Studies
Byte-Level Static Analysis	Raw bytes, byte n-grams	Enables ultra-fast pre-execution screening; avoids disassembly overhead	Highly sensitive to packing and syntactic transformations	[15], [29]
Opcode Sequence Modeling	Opcode n-grams, opcode sequences	Captures instruction-level semantics; resilient to superficial obfuscation	Vulnerable to heavy instruction substitution and reordering	[27], [28]
Deep Opcode Representation Learning	LSTM, CNN, self-attention over opcode streams	Learns long-range dependencies; rivals dynamic analysis accuracy	Increased computational cost; limited interpretability	[26]
Static Behavioral Encoding	RGB image encodings of static reports	Enables transfer learning and Vision Transformers; strong generalization	Encoding design affects robustness	[16]
API Import and Structural Analysis	Imported APIs, function statistics	Lightweight and interpretable; suitable for endpoint deployment	Lacks runtime execution context	[14]
Permission-Based Static Analysis (Mobile)	Requested permissions, application metadata	Effective first-line defense; low overhead	High false positives for benign applications	[34]
Structural Binary Feature Analysis	Import Address Table (IAT), PE headers	Strong for ransomware family clustering and attribution	Limited behavioral expressiveness	[30]
Multi-Perspective Static Similarity Analysis	Hybrid binary and visualization features	Robust against packing and polymorphism	Increased system complexity	[31]
Graph-Based Static Modeling	Feature dependency graphs, GNN representations	Captures relational semantics; supports early-stage detection	Graph construction overhead	[17]
Optimized Static Feature Selection (IoT)	Reduced static feature subsets	Resource-efficient; suitable for constrained environments	Dataset- and optimizer-dependent	[32]

3.3 Static Analysis Surveys and Foundational Studies

Several surveys and foundational studies provide essential background for static ransomware detection. Gregory Paul and Gireesh Kumar [35] present an early framework comparing static and dynamic malware analysis, emphasizing behavioral artifact extraction via sandboxing. Although ransomware is not the primary focus, this work establishes methodological foundations that underpin later static and hybrid ransomware detection research.

Critical synthesis. Taken together, the static-analysis literature reveals a recurring and frequently under-acknowledged tension. The headline accuracies reported for opcode- and image-based representations, frequently exceeding 99%, are obtained under closed-world assumptions—fixed family sets, balanced classes, and single-snapshot binaries—that rarely survive contact with packed, polymorphic, or previously unseen samples. Static detection is therefore best positioned not as a standalone solution but as a fast, low-cost pre-filter whose principal value is triage, narrowing the candidate space before costlier dynamic analysis is invoked. Two gaps stand out. First, few static studies report robustness under adversarial packing or controlled obfuscation [10], [36], leaving their resilience claims largely untested. Second, temporal and cross-family generalization—training on families seen before a cutoff and testing on those appearing afterward [9]—is almost never evaluated. These two omissions, rather than raw accuracy, are arguably the decisive criteria for judging static methods in deployment.

To consolidate the theoretical foundations, feature abstractions, and representative static ransomware detection techniques discussed in this section, Table I provides a structured comparison of major static analysis categories, their core feature representations, theoretical strengths, inherent limitations, and supporting studies.

4 DYNAMIC BEHAVIOR–BASED RANSOMWARE DETECTION

Dynamic analysis captures runtime behavioral semantics that remain inaccessible to purely static inspection, exposing malicious intent as ransomware interacts with operating-system services, storage, memory, and networks. It is therefore especially suited to modern ransomware that uses packing, polymorphism, delayed execution, and environment-aware logic to defeat static inspection. Whatever the delivery vector, crypto-ransomware, lockers, ransomworms, and fileless variants must ultimately perform an observable sequence of actions—process initialization, file enumeration, cryptographic setup, registry modification, memory allocation, and network or lateral-movement activity—and these traces form the basis for detection.

Execution can be decomposed into behavioral phases—environment probing, persistence, key generation and cryptographic preparation, file-system traversal and encryption, and optional command-and-control interaction—and effective detection aims to flag intent as early in this lifecycle as possible, ideally before irreversible encryption. The temporal structure of behavior (which actions occur, in what order, and within what window) is thus central to dynamic detection.

Dynamic signals span complementary abstraction layers: high-level API calls and execution traces are interpretable but platform-specific; low-level system calls, storage I/O, memory artifacts, and hardware performance counters resist evasion at higher cost; and network-centric signals extend visibility beyond the host toward pre-encryption detection. This defense-in-depth view is constrained by intrinsic limitations—sandbox evasion, delayed activation, partial observability under short execution windows, and runtime overhead and privacy—while the non-stationary nature of ransomware behavior induces concept drift that demands adaptive, streaming-capable learning.

Dynamic detection therefore plays three roles in modern defenses: behavior-driven identification of obfuscated and zero-day ransomware beyond the reach of static methods; provision of the temporal and contextual intelligence required for early-stage, pre-encryption intervention; and integration of static, network, memory-forensic, and recovery components within hybrid, multi-tier frameworks. The following subsections review API-level execution traces, deep sequence modeling, explainable behavioral analysis, low-level system and hardware signals, network-centric detection, hybrid architectures, and drift-aware pre-encryption frameworks.

4.1 API Call and Execution Trace–Based Detection

Dynamic ransomware detection exploits runtime behavioral traces collected during program execution, with API call sequences emerging as one of the most informative and widely adopted feature sources. Unlike static analysis,

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 06 , June - 2026

API-based dynamic approaches capture temporal execution semantics, enabling reliable detection of obfuscated, packed, and environment-aware ransomware variants.

4.1.1 Classical Machine Learning Approaches

Early dynamic ransomware detection research predominantly relied on classical machine learning (ML) classifiers trained on API call frequencies, ordered sequences, or statistical representations derived from execution traces. These studies collectively established that carefully engineered behavioral features can achieve high detection accuracy with relatively low computational overhead, making them suitable for real-time deployment.

Almousa et al. [37] propose an API-driven detection framework that models the ransomware execution lifecycle on Windows systems using sandboxed runtime traces. By extracting API call behavior and optimizing multiple ML classifiers via grid search, the framework achieves 99.18% detection accuracy, demonstrating the strong discriminative power of API-level behavioral monitoring and motivating its integration into multi-layer defense architectures.

Nguyen and Lee [3] show that tree-based ensemble methods—particularly LightGBM—are highly effective for ransomware detection using API call sequences. Their multiclass model, evaluated across eight ransomware families and benign applications, achieves 98.7% overall accuracy with near-perfect detection of WannaCry and FileCoder variants, highlighting that ensemble learning can rival deep learning performance at significantly lower computational cost.

Feature dimensionality reduction has been addressed by Kader et al. [38], who formulate ransomware detection as a binary classification problem using dynamic runtime features. By comparing manual feature engineering with automated K-best feature selection, the study demonstrates that dimensionality can be substantially reduced without degrading detection accuracy, a critical consideration for latency-sensitive and resource-constrained deployments.

Several studies emphasize that temporal ordering of API calls is essential for early and reliable detection. Sukul et al. [5] introduce an augmented bootstrapping approach for early-stage detection under partial observability. Using timestamped API *n*-grams (bi-gram to 5-gram) with TF–IDF weighting, their framework achieves high accuracy while reducing false positives, making it particularly suitable for pre-encryption detection scenarios.

Chen et al. [39] focus on automated extraction and ranking of discriminative API-based behavioral patterns from large-scale sandbox logs. By combining TF–IDF weighting, Fisher's Linear Discriminant Analysis, and Extra Trees classifiers, the framework enables effective early-stage detection across multiple ransomware families, including WannaCry, GandCrab, and Locky, while also enhancing forensic interpretability and reducing analyst workload.

Family-specific investigations further validate the effectiveness of classical ML-based API monitoring. Usharani et al. [40] analyze the GandCrab ransomware family using runtime behavioral features such as IP addresses, URLs, and file attributes. An ensemble of Random Forest, Gradient Boosting, and Support Vector Machine classifiers achieves 98.45% accuracy, confirming that classical ML techniques remain effective for targeted crypto-ransomware detection.

More evasive ransomware variants are addressed by Ahmed et al. [41], who propose a dynamic detection framework targeting high-survivability ransomware that employs packing and obfuscation. By integrating multiple runtime behavioral feature sets with TF–IDF–based feature selection, the system achieves an area under the curve (AUC) of 0.987 with an exceptionally low false positive rate, underscoring the importance of curated behavioral features in adversarial settings.

The lack of standardized datasets has historically limited reproducibility in dynamic ransomware research. Herrera Silva and Hernández-Alvarez [42] mitigate this issue by releasing a public dataset derived from sandbox execution of encryptor and locker ransomware. The dataset contains 50 carefully selected runtime features and enables ML models to achieve detection accuracies exceeding 99%, significantly improving comparability across studies.

Recent work has also introduced explainability into API-based dynamic detection. Gulmez et al. [20] propose XRan, an explainable framework that models API calls, dynamic library activity, and mutex operations while preserving sequence relationships. Although discussed further in the context of deep sequence modeling, XRan demonstrates that explainability mechanisms such as SHAP and LIME can substantially enhance trust and transparency in dynamic ransomware detection systems.

4.1.2 Deep Sequence Models (LSTM, Transformer, and Hybrid Architectures)

While classical machine learning approaches demonstrate strong performance using engineered behavioral features, they are often limited in modeling long-range temporal dependencies and complex execution semantics. To overcome these limitations, recent research increasingly adopts deep sequence learning models, treating ransomware execution traces as ordered sequences analogous to natural language.

One of the earliest applications of deep sequence modeling for ransomware detection is presented by Maniath et al. [18], who introduce a Long Short-Term Memory (LSTM)–based framework that models API call sequences extracted during dynamic execution. By representing API calls as word tokens, the LSTM captures temporal dependencies that distinguish ransomware from benign software. This work establishes sequence-aware learning as a foundational paradigm for scalable and automated dynamic ransomware detection.

Building on this paradigm, Jahromi et al. [4] propose an enhanced stacked LSTM architecture designed for time-critical and safety-critical environments. Unlike conventional LSTM models with random initialization, their approach employs unsupervised layer-wise pretraining to improve convergence stability and robustness. Evaluated across Windows malware, ransomware, Internet of Things (IoT), and Android datasets, the model achieves 99.1% accuracy with strong Matthews Correlation Coefficient (MCC) and area under the curve (AUC) scores, demonstrating suitability for real-time ransomware detection.

Hybrid deep learning frameworks further improve robustness by integrating multiple behavioral perspectives. Shaukat and Ribeiro [12] propose RansomWall, a layered defense system that combines static inspection, dynamic behavior monitoring, and machine learning–based classification. Although not purely sequence-based, RansomWall incorporates early behavioral monitoring and runtime feature learning, achieving a 98.25% detection rate with near-zero false positives and demonstrating strong zero-day detection capability in Windows environments.

Transformer-based architectures have emerged as a powerful alternative to recurrent models for sequence modeling. Youssef et al. [43] apply Bidirectional Encoder Representations from Transformers (BERT) to ordered API call sequences extracted from dynamic execution logs. Trained on more than 93,000 Portable Executable (PE) samples, the

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 06 , June - 2026

model achieves robust performance across malware categories, including ransomware, with an F1-score of 0.85, highlighting the effectiveness of attention mechanisms in capturing long-range execution dependencies.

Hybrid convolutional neural network (CNN)–Transformer architectures further enhance representation learning. Chen et al. [19] propose RANsomCheck, which combines CNN-based local pattern extraction with Transformer-based global sequence modeling using novel semantic encodings of API call sequences. The framework achieves 99.94% accuracy and 100% recall, demonstrating exceptional robustness against evolving ransomware variants and highlighting the operational viability of deep sequence models for real-time defense.

Comparative analyses reinforce the dominance of sequence learning paradigms. Lin et al. [44] evaluate LSTM and BERT models for ransomware detection using API call sequences treated as textual data. Both models achieve approximately 95% accuracy, with BERT excelling in long-range dependency modeling and LSTM offering lower computational overhead, illustrating the trade-off between expressiveness and efficiency.

Deep sequence modeling has also been extended beyond API calls. Nahmias et al. [45] introduce TrustSign, which extracts deep representations from volatile memory snapshots using transfer learning from a pre-trained VGG-19 network. This approach enables detection of fileless ransomware and generates robust behavioral signatures with 99.5% accuracy, particularly suited for trusted cloud and virtualized environments.
Similarly, Alvi et al. [46] propose RGV2, which models file input/output behavior using FastText embeddings learned from Event Tracing for Windows (ETW) logs and classifies them using one-dimensional convolutional networks, achieving 99.64% accuracy with low false-positive rates.

To reduce detection latency, temporal windowing strategies have been explored. Iwan et al. [47] propose a real-time host-based detection framework using sliding windows over API call sequences combined with term frequency–inverse document frequency (TF–IDF) and *n*-gram representations. Their system detects ransomware within fewer than 40,000 API calls, significantly before full encryption begins.

Finally, sequence modeling at lower abstraction levels has been investigated. von der Assen et al. [48] propose HyperDtct, a hypervisor-based ransomware detection framework that collects system calls via virtual machine introspection. By operating outside the guest operating system, HyperDtct resists malware tampering and achieves an F1-score of 0.97 with detection within 10 seconds, making it well suited for cloud and virtualized environments.

4.1.3 Explainable and Interpretable Dynamic Detection

As dynamic ransomware detection systems increasingly rely on complex deep learning models, their lack of transparency has emerged as a major barrier to adoption in security-critical environments. Explainable and interpretable approaches aim to provide human-understandable justifications for detection decisions, supporting trust, accountability, and actionable incident response.

Tan and Abdulnabi [49] propose TriFRIM, a triple-filter ransomware detection framework that integrates signature-based, network-based, and behavioral features into a unified and interpretable architecture. Rather than optimizing solely for detection accuracy, TriFRIM emphasizes analyst-oriented reasoning and multi-view decision support, highlighting the importance of holistic interpretability in ransomware defense systems.

Explainability has also been explored at the feature engineering level. Mowri et al. [21] compare iterative feature selection using Recursive Feature Elimination with Cross-Validation (RFECV) against explainability-driven feature ranking using SHapley Additive exPlanations (SHAP). Their analysis shows that RFECV often fails to identify semantically meaningful features and can increase false alarms, whereas SHAP-based ranking yields more stable and interpretable feature subsets under dynamic conditions.

A comprehensive model-agnostic explainable framework is introduced by Alvi and Jalil [22] through XRGuard. The system leverages ETW logs to monitor file input/output behavior and applies SHAP and Local Interpretable Model-agnostic Explanations (LIME) to explain detection outcomes at both global and instance levels. XRGuard achieves 99.69% detection accuracy with a very low false positive rate while providing explanations suitable for security operations centers.

Explainability considerations are also reflected in cloud-based detection systems. Nguyen et al. [50] propose a hybrid deep learning framework for cloud environments that combines static attributes with dynamic system-call behavior and addresses class imbalance using Conditional Tabular Generative Adversarial Networks (CTGANs). Although performance-driven, the structured feature design and explicit behavioral modeling enhance interpretability and post-detection analysis in shared cloud infrastructures.

Table II summarizes representative dynamic, behavior-based ransomware detection approaches, organizing them by behavioral layer and modeling paradigm together with the representative studies that exemplify each.

4.2 System Call, Storage, File-System, and Memory-Level Behavior

While API call sequences provide a high-level view of ransomware execution, advanced ransomware increasingly bypasses API-level monitoring through direct system calls, low-level file-system manipulation, and kernel-level evasion techniques. Consequently, recent research has shifted toward lower-layer behavioral signals, including system calls, storage access patterns, file-system events, and hardware-level execution traces. These signals are inherently more difficult to obfuscate and have proven particularly effective for early-stage and zero-day ransomware detection.

4.2.1 System Call and Storage-Level Detection

One of the earliest demonstrations of storage-centric ransomware detection is presented by Hirano and Kobayashi [51], who introduce a dynamic detection framework based on storage access patterns collected via a live-forensic hypervisor. By extracting only five compact input/output–related features and applying classical machine learning classifiers such as Random Forest, Support Vector Machines, and k-nearest neighbors, the framework achieves an F-measure of 98%. This study establishes that low-level disk behavior alone can reliably distinguish ransomware from benign workloads, even when higher-level behavioral indicators are concealed or delayed. To address reproducibility challenges in storage-level ransomware research, Hirano et al. [52] later introduce RanSAP, an open dataset capturing ransomware storage access patterns under diverse operating conditions, including multiple operating systems and full-disk encryption. In addition to dataset construction, the authors provide baseline feature extraction and detection pipelines, enabling controlled and comparable evaluation of storage-centric ransomware detectors. RanSAP has since become a foundational reference for research on storage-level behavioral modeling.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 06 , June - 2026

TABLE II. DYNAMIC RANSOMWARE DETECTION: BEHAVIORAL LAYERS, MODELING PARADIGMS, AND REPRESENTATIVE STUDIES.

Behavioral Layer	Runtime Artifacts Observed	Learning Paradigm	Detection Objective	Key Strengths	Limitations	Representative Studies
API Call Frequency & Statistics	API counts, execution summaries	Classical ML (RF, SVM, LightGBM)	Post-execution detection	Low overhead, interpretable	Vulnerable to delayed execution	. [37], [3]
Ordered API Call Sequences	Timestamped API n-grams	TF–IDF + ML	Early-stage / pre-encryption detection	Temporal awareness, efficient	Limited long-range context	[5], [39]
Family-Specific Behavioral Traces	Network, file, and execution features	Ensemble ML	Targeted family detection	High precision	Limited generalization	[40]
Obfuscation-Resilient Runtime Features	Multi-view runtime behavior	TF–IDF + ML	Detection of evasive ransomware	Robust to packing	Feature engineering complexity	[41]
Public Dynamic Datasets	Sandbox-extracted runtime features	Benchmarking	Reproducibility	Enables fair comparison	Dataset bias	[42]
Sequential Deep Learning	API call sequences	LSTM	Behavioral sequence modeling	Captures temporal dependencies	Training cost	[18], [4]
Hybrid Deep Architectures	API calls + runtime features	CNN–Transformer	Early and robust detection	Long-range modeling	Computational overhead	[19]
Transformer-Based Models	Long execution traces	BERT	Context-aware detection	Strong generalization	Resource intensive	[43], [44]
Memory-Centric Deep Learning	Volatile memory snapshots	CNN + Transfer Learning	Fileless ransomware detection	Resilient to evasion	Requires memory access	[45]
File I/O Semantic Modeling	ETW file I/O events	Embedding + CNN	Encryption-phase detection	Semantic generalization	Platform-specific	[46]
Sliding Window Early Detection	Partial execution traces	N-grams + TF–IDF	Pre-encryption detection	Low latency	Partial observability	[47]
Hypervisor-Based Monitoring	System calls via VMI	ML	Anti-tampering detection	OS-independent	Deployment complexity	[48]
Explainable Dynamic Detection	Runtime + explanation vectors	SHAP, LIME	Trustworthy detection	Analyst transparency	Extra computation	[20], [49], [22]
Drift-Aware Cloud Detection	Static + dynamic cloud logs	Deep Learning + CTGAN	Long-term robustness	Handles imbalance	Infrastructure dependent	[50]

File-system behavior has also been examined at a lower abstraction level using input/output request packets (IRPs). Ayub and Siraj [53] conduct a large-scale empirical study analyzing IRP sequences from 383 ransomware samples spanning 21 families. By evaluating both supervised ensemble classifiers and one-class anomaly detection models, the study demonstrates that ransomware can be detected based solely on execution-time I/O behavior, independent of API-level semantics. Although longer observation windows are sometimes required, IRP-based detection shows strong generalization to previously unseen ransomware families, making it well suited for zero-day scenarios.

Beyond software-level monitoring, several studies leverage hardware performance counters (HPCs) to capture execution behavior at the microarchitectural level. Mohan Anand et al. [23] introduce HiPeR, a hardware-assisted ransomware detection framework that identifies ransomware within 3–4 seconds of execution, enabling termination before encryption completes. Because HPCs are largely operating-system–independent and difficult for malware to manipulate, HiPeR demonstrates strong resilience against obfuscation and system-call–level evasion.

Deep learning has further enhanced hardware-level analysis. Ganfure et al. [24] propose DeepWare, which transforms HPC distributions into image-like representations and applies convolutional neural networks for classification. Using execution snapshots as short as 100 ms, DeepWare achieves 98.6% recall with near-zero false positives and exhibits strong generalization to previously unseen ransomware families, highlighting the effectiveness of representation learning at the hardware level.

Complementary evidence is provided by Aurangzeb et al. [54], who explore ransomware detection using hardware execution profiles derived from performance counters on Microsoft Windows systems. Evaluating multiple machine learning classifiers, the study reports an F-measure of 0.97, confirming that hardware-level metrics offer a robust and evasive-resistant signal, particularly against heavily obfuscated ransomware samples.

Scalability and deployment efficiency are addressed by Woralert et al. [55], who propose a Light Gradient Boosting Machine (LightGBM)–based framework for malware detection using performance monitoring counters. The framework achieves 99.95% binary classification accuracy and supports rapid transfer learning to adapt to new ransomware variants. By avoiding reliance on intrusive software instrumentation, the approach is well suited for cloud platforms and high-performance computing environments.

Finally, Pan and Shu [56] introduce a hardware-assisted ransomware detection framework enhanced with automated machine learning (AutoML). By combining microprocessor performance monitoring with neural architecture search and adversarial training, the system dynamically optimizes detection models while improving robustness against evasion. The proposed framework achieves up to a 4.8× reduction in detection latency alongside significant accuracy gains, demonstrating the potential of automated and adaptive hardware-assisted detection for real-time ransomware defense.

4.2.2 Registry, File Entropy, and File-System Encryption Analysis

Beyond system calls and storage access patterns, ransomware exhibits distinctive behavioral footprints in Windows registry manipulation, file-system entropy evolution, and encryption workflows. These artifacts provide complementary detection surfaces that are particularly effective for distinguishing malicious encryption activity from legitimate cryptographic operations, a long-standing challenge in dynamic ransomware detection.

Jethva et al. [7] propose a multilayer behavioral detection framework that jointly analyzes grouped registry key operations, file entropy variation, and file signature changes.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 06 , June - 2026

By correlating registry manipulation patterns with entropy-driven file modifications, the framework effectively differentiates ransomware-triggered encryption from benign user-initiated encryption. Evaluated on a dataset comprising 666 ransomware and 103 benign samples, the approach achieves high detection accuracy using Support Vector Machines, logistic regression, and Random Forest classifiers. This work demonstrates that multi-dimensional behavioral fusion significantly improves robustness against obfuscation and encryption-based evasion.

Graph-based modeling has been explored to enhance interpretability of file-system behavior. Rosli et al. [57] introduce a graph-theoretic representation in which ransomware file activities are modeled as behavioral graphs stored in a Neo4j database. By visualizing relationships among file operations, the framework enables analysts to identify influential behavioral nodes and attack pathways. Although evaluated on a limited dataset, the study highlights the value of graph-based behavioral abstraction as a complementary tool for forensic analysis and attack understanding rather than standalone detection.

Recent research has moved beyond static entropy thresholds to address adaptive and intermittent encryption strategies. Mahboubi et al. [58] target ransomware that deliberately avoids sustained high-entropy writes to evade conventional entropy-based detectors. The authors propose an adaptive file-system–level detection framework based on statistical storage attributes, including write-buffer block characteristics, combined with online learning using Hoeffding Trees. Evaluated on 11,928 encrypted files from 75 ransomware families, the approach demonstrates strong resilience against partial and evolving encryption behaviors, making it particularly suitable for real-time and adaptive threat environments.

Memory-aware detection further complements file-system analysis. Aljabri et al. [25] investigate memory forensics–driven ransomware detection by analyzing volatile memory dumps from modern ransomware families such as REvil, LockBit, and BlackCat. Using a compact set of memory-resident features, an Extreme Gradient Boosting (XGBoost) classifier achieves 97.85% accuracy with a 2% false positive rate. The study highlights that memory artifacts—including injected code, runtime encryption buffers, and process structures—remain observable even when file-system and API-level indicators are deliberately obfuscated.

Finally, ransomware detection must be considered within the broader incident response lifecycle. Upadhyay et al. [59] provide a comprehensive survey of ransomware detection and data recovery strategies, reviewing behavioral, signature-based, and machine learning–driven approaches alongside recovery mechanisms such as backups and decryption tools. While not proposing a new detection model, the study reinforces the practical importance of file-system–centric detection as an enabler for timely containment and recovery planning.

4.2.3 Memory Feature–Based Detection and Volatile Forensics

Memory-based analysis provides a powerful and evasive-resistant detection surface for ransomware, particularly against samples that employ heavy obfuscation, delayed execution, or direct system-call invocation to bypass API- and file-system–level monitoring. Unlike static binaries or transient behavioral logs, volatile memory captures runtime state, including decrypted code, injected payloads, encryption buffers, and process-level artifacts that are difficult for ransomware to conceal.

Arfeen et al. [60] propose a process-centric volatile memory forensics framework that overcomes the limitations of single-snapshot analysis by acquiring multiple time-spaced memory dumps during execution. By extracting temporal runtime features from evolving memory states, machine learning classifiers achieve improved discrimination between benign and ransomware processes. This approach is particularly valuable for post-infection analysis, severity assessment, and attribution, especially when ransomware exhibits dormant or partial execution behavior.

Bridging memory forensics with representation learning, Ali-Gombe et al. [61] introduce cRGB_Mem, a framework that transforms in-memory allocation patterns into RGB images and applies convolutional neural networks for classification. Operating entirely on post-execution memory artifacts, cRGB_Mem enables detection even when behavioral hooks are bypassed. The method achieves 95.98% accuracy on known samples and demonstrates robustness against obfuscated malware, illustrating the potential of memory imaging and visual learning for ransomware detection.

Although primarily associated with network-centric analysis, Ganame et al. [62] provide complementary insight by demonstrating that network-level anomalies can precede observable file-system or memory artifacts. Their case study on WannaCry shows that early network indicators can signal ransomware activity before payload execution. As discussed in Section IV-C, this observation reinforces the importance of cross-layer correlation, where early network signals can be integrated with memory forensics for comprehensive detection and investigation.

4.3 Network Traffic, Honeypots, and IDS-Based Detection

Network-centric ransomware detection exploits the observation that many ransomware families establish command-and-control (C2) communication, reconnaissance activity, or lateral movement before or during payload execution. Unlike host-based techniques, network-level analysis enables pre-encryption and pre-compromise detection without requiring endpoint instrumentation, making it particularly attractive for enterprise-scale ransomware defense. As a result, network traffic analysis, honeypots, and intrusion detection systems (IDS) play a critical role in early warning and coordinated response.

Almashhadani et al. [63] present one of the earliest comprehensive studies on network-based crypto-ransomware detection, using Locky ransomware as a case study. The authors design a multi-classifier IDS operating at both packet and flow levels, with parallel classifiers analyzing complementary network features. Experimental results demonstrate high detection accuracy and low false positive rates, confirming that ransomware-related network activity can be identified before file encryption begins, thereby establishing network monitoring as an effective early-stage detection layer.

To address zero-day ransomware and evolving attack behavior, Zuhair et al. [64] propose a multi-tier streaming analytics framework that continuously learns from dynamic data streams. By combining static and dynamic traits across approximately 40,000 ransomware, malware, and benign samples spanning 14 ransomware families, the system achieves an average accuracy of 97%. Its ability to model variant evolution and ancestral relationships makes it well suited for long-term deployment in environments subject to concept drift.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 06 , June - 2026

TABLE III. : LOW-LEVEL HOST AND NETWORK-CENTRIC DYNAMIC RANSOMWARE DETECTION TECHNIQUES

Detection Layer	Behavioral Artifacts	Monitoring Level	Learning Paradigm	Detection Objective	Key Strengths	Limitations	Representative Studies
System Call Behavior	Direct system calls	OS kernel	ML classifiers	Runtime detection	Bypasses API evasion	OS-dependent	Hirano and Kobayashi [51]
Storage Access Patterns	Disk I/O operations	Hypervisor	ML	Pre-encryption detection	OS-independent	Deployment complexity	Hirano et al. [52]
IRP-Based File-System Traces	I/O request packets	Kernel driver	Ensemble ML, Anomaly Detection	Zero-day detection	Family generalization	Longer observation windows	Ayub and Siraj [53]
Hardware Performance Counters	Microarchitectural events	CPU hardware	ML	Ultra-early detection	Strong evasion resistance	Hardware dependency	Mohan Anand et al. [23]
HPC Imaging & Deep Learning	HPC distributions	CPU hardware	CNN	Zero-day detection	High recall	Feature interpretability	Ganfure et al. [24]
HPC-Based ML Profiling	Execution profiles	CPU hardware	ML	Obfuscated ransomware detection	OS-independent	Architecture variability	Aurangzeb et al. [54]
Scalable HPC Detection	Performance counters	Cloud / HPC systems	LightGBM + Transfer Learning	Large-scale deployment	High scalability	Limited semantic insight	Woralert et al. [55]
AutoML Hardware Detection	HPC + model search	CPU hardware	AutoML + Adversarial Training	Adaptive early detection	Reduced latency	System complexity	Pan and Shu [56]
Registry & Entropy Analysis	Registry keys, entropy	OS-level	ML	Encryption discrimination	High interpretability	Windows-specific	Jethva et al. [7]
File-System Graph Modeling	File operation graphs	OS-level	Graph analysis	Forensic analysis	Visual interpretability	Limited scalability	Rosli et al. [57]
Adaptive Encryption Detection	Storage statistics	OS-level	Online Learning	Evasive encryption detection	Drift resilience	Feature tuning	Mahboubi et al. [58]
Volatile Memory Forensics	Memory dumps	RAM	ML	Post-compromise detection	Obfuscation resistance	High overhead	Aljabri et al. [25]
Temporal Memory Profiling	Multi-snapshot memory	RAM	ML	Dormant ransomware detection	Temporal insight	Acquisition cost	Arfeen et al. [60]
Memory Imaging + CNN	Heap allocation images	RAM	CNN	Fileless ransomware	Bypasses hooks	Post-execution only	Ali-Gombe et al. [61]
Network Flow & Packet Analysis	Packet/flow statistics	Network	IDS + ML	Pre-encryption detection	No endpoint instrumentation	Encrypted traffic	Almashhadani et al. [63]
Streaming Network Analytics	Live traffic streams	Network	Adaptive ML	Zero-day detection	Drift handling	Resource intensive	Zuhair et al. [64]
Honeypot-Based Detection	Attack interaction logs	Network	Voting ML	Early warning	High confidence	Deployment effort	Ng et al. [65]
Encrypted Traffic Modeling	TLS metadata	Network	Transformer	Pre-compromise detection	Encryption-agnostic	Feature abstraction	Yang et al. [66]
Enterprise Log Analysis	AD event logs	Network/domain	ML + N-grams	Lateral movement detection	Enterprise relevance	Domain-specific	Bhandary and Nicholas [67]
APT Correlation Analysis	TTP sequences	Network	Analytical modeling	Attribution	Strategic insight	Not real-time	Ali Khan and Almulhem [68]
Cross-Layer Network–Host Fusion	Network + memory + registry	Hybrid	ML	Fileless ransomware	Defense-in-depth	System complexity	Bandara et al. [69]
Explainable Network-Aware Detection	ETW + network logs	Hybrid	XAI	SOC usability	Trust & transparency	Runtime overhead	XRGuard [22]
Mobile Network-Based Detection	Network + behavior	Mobile	ML	Android ransomware	Lightweight	Lower accuracy	Arputha Rathina et al. [70]

Honeypot-based detection provides an alternative network-centric perspective. Ng et al. [65] introduce VoterChoice, a framework that integrates client honeypots, intrusion prevention systems, and multiple machine learning classifiers using a voting mechanism. By aggregating decisions from diverse detection models, VoterChoice improves robustness and confidence compared to single-classifier systems, demonstrating suitability for enterprise-scale ransomware monitoring and prevention.

With the widespread adoption of encrypted communication by ransomware operators, detecting malicious behavior in encrypted traffic has become increasingly important. Yang et al. [66] propose PETNet, a Transformer-based framework for identifying Cobalt Strike HTTPS traffic, a tool frequently abused in ransomware C2 infrastructures. By modeling encrypted traffic metadata and temporal patterns, PETNet maintains robustness against concept drift over a 14-month evaluation period, demonstrating that encrypted traffic analysis

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 06 , June - 2026

can function as a pre-compromise ransomware detection layer independent of endpoint artifacts.

Enterprise environments introduce additional challenges due to complex authentication and privilege management mechanisms. Bhandary and Nicholas [67] conduct an in-depth behavioral analysis of ransomware activity in Active Directory environments, focusing on BlackMatter, Conti, LockBit, and Midnight ransomware families. Using Windows Event Logs collected from a virtualized domain infrastructure, the study identifies consistent behavioral patterns such as service manipulation and credential abuse through *n*-gram modeling. These findings provide a strong foundation for domain-level and rule-assisted machine learning detection in organizational networks.

Beyond detection, network-centric analysis also supports attribution and threat intelligence correlation. Ali Khan and Almulhem [68] analyze ransomware-driven advanced persistent threat (APT) campaigns by correlating behavioral indicators, propagation mechanisms, and tactics, techniques, and procedures (TTPs), using the Black Basta campaign as a case study. While not a direct detection framework, this work complements network-based ransomware defense by enabling context-aware classification and strategic response.

Modern ransomware increasingly employs fileless techniques that rely on network-based delivery and in-memory execution. Bandara et al. [69] propose STEALTH EYE, a real-time behavioral detection framework that combines network communication analysis with host-level monitoring of memory activity, dynamic-link library injection, registry changes, and cryptographic API usage. By analyzing behavior in 60-second windows, the system detects fileless malware—including ransomware—before significant damage occurs, highlighting the importance of cross-layer host–network correlation.

Although primarily host-focused, XRGuard [22] is also relevant in network-centric contexts, as its explainable architecture can be extended to incorporate network-derived Event Tracing for Windows events. By combining model-agnostic detection with SHAP and LIME explanations, XRGuard demonstrates how interpretability enhances trust and usability in security operations centers where network alerts must be rapidly triaged.

Finally, ransomware detection in mobile and resource-constrained environments has also leveraged network-aware features. Arputha Rathina et al. [70] propose a hybrid machine learning framework for Android ransomware detection that integrates static attributes, dynamic behavior, and threat-text analysis. Although detection accuracy is lower than desktop-focused approaches, the study demonstrates that network and behavioral signals remain viable for mobile ransomware detection where traditional endpoint defenses are limited.

To consolidate lower-layer host-based and network-centric dynamic ransomware detection techniques, Table III summarizes the behavioral abstraction level, monitored artifacts, learning paradigms, detection objectives, and representative studies.

5 Hybrid Static–Dynamic and Multi-Tier Frameworks

Purely static or purely dynamic ransomware detection approaches face inherent limitations when deployed in isolation. Static methods struggle with heavy obfuscation and packing, while dynamic techniques are vulnerable to sandbox evasion, delayed execution, and partial observability. To address these challenges, a growing body of research adopts hybrid static–dynamic and multi-tier architectures, integrating complementary feature sources, detection layers, and decision strategies.

Early work in this direction is represented by RansHunt, proposed by Hasan and Rahman [13]. RansHunt integrates static binary features with dynamic behavioral indicators and employs Support Vector Machines for classification. By fusing features from both domains, the framework improves detection accuracy compared to single-view approaches, particularly against IDS-aware and sandbox-evasive ransomware.

Layered defense architectures extend hybridization beyond feature fusion. Shaukat and Ribeiro [12] introduce RansomWall, a multi-layer ransomware defense system combining static inspection, dynamic behavior monitoring, and machine learning–based classification. A distinctive feature of RansomWall is its trap layer, which enables early detection while backing up suspicious file modifications to prevent irreversible data loss. Evaluated on 574 ransomware samples from 12 families, the framework achieves a 98.25% detection rate with near-zero false positives, demonstrating strong zero-day capability.

Hybrid frameworks have also been extended to include response and recovery logic. Pahuja et al. [71] propose RansomShield, an integrated architecture that combines network segmentation, honeypots, and machine learning–based traffic analysis with automated data recovery. Logs collected from honeypots are analyzed using multiple classifiers, with XGBoost outperforming artificial neural networks and Support Vector Machines for early detection. By coupling detection with recovery, RansomShield emphasizes operational resilience rather than detection alone.

Network-centric hybrid systems target the propagation phase of ransomware. Almashhadani et al. [72] propose MFMCNS, a multi-feature, multi-classifier network-based system designed to detect ransomworm propagation during lateral movement. By analyzing session-based and time-based network flows and applying decision fusion across parallel classifiers, MFMCNS enables early containment before payload execution.

Several studies adopt a general hybrid machine learning framework perspective. Mumtaz et al. [73] present a broad malware detection framework emphasizing ransomware among other threat classes. By combining features from system calls, network traffic, and file behavior, and evaluating classifiers such as Decision Trees, Support Vector Machines, and Random Forests, the study reinforces the importance of hybrid feature engineering for reducing false positives.

Deep learning–driven hybridization across platforms is exemplified by SwiftR, proposed by Karbab et al. [74]. SwiftR integrates static intermediate-representation features with dynamic behavioral word embeddings using a hierarchical neural architecture. Static analysis is performed first, while an LSTM-based dynamic module is invoked only when confidence is low. Evaluated on over 40,000 ransomware samples, SwiftR achieves up to 98% F1-score and maintains performance under unknown-family scenarios, representing a landmark contribution toward scalable and portable hybrid detection.

Hybrid frameworks increasingly incorporate streaming analytics and concept drift adaptation. Ceschin et al. [8] model malware detection as an evolving data stream and demonstrate that adaptive classifiers and drift detectors significantly outperform static pipelines on long-term Android datasets. Similarly, Fernando and Komninos [9] propose FeSAD, a ransomware detection framework with explicit drift calibration and decision layers that dynamically adapt feature selection and classification strategies to sustain performance over time.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 06 , June - 2026

Hybridization is also critical in resource-constrained environments. Abid et al. [75] introduce ECMT, a framework that integrates in-memory forensic attributes with ensemble learning for Internet of Things malware detection, including ransomware. By combining Support Vector Classifiers, Quadratic Discriminant Analysis, AdaBoost, and neural models, ECMT achieves up to 96% accuracy while addressing scalability and concept drift.

Finally, datasets play a foundational role in hybrid research. RanSAP, introduced by Hirano et al. [52], provides an open dataset of ransomware storage access patterns collected via hypervisor-based monitoring, along with a baseline detection pipeline. RanSAP enables reproducible evaluation of hybrid behavioral models that integrate storage-centric and higher-level features.

5.1 Concept Drift, Streaming Analytics, and Pre-Encryption Detection

Ransomware detection systems operate in highly non-stationary threat environments, where attacker tactics, payload structures, and execution behaviors evolve continuously. This phenomenon, commonly referred to as concept drift, gradually degrades the effectiveness of static feature sets and fixed classifiers. In parallel, the need for pre-encryption detection has become critical, as post-encryption identification often fails to prevent irreversible data loss. Consequently, recent research increasingly emphasizes drift-aware learning, streaming analytics, and ultra-early detection mechanisms designed for long-term operational resilience.

Fernando and Komninos [9] introduce FeSAD, a ransomware detection framework explicitly designed to address concept drift without frequent retraining. FeSAD integrates feature selection, drift calibration, and drift decision layers, allowing the system to adapt dynamically under multiple drift scenarios. Rather than optimizing short-term accuracy, the framework prioritizes detection stability over time, establishing FeSAD as a reference architecture for production-grade ransomware defense in evolving environments.

Complementing classifier-level adaptation, Fernando and Komninos [76] address concept drift at the feature engineering stage through FeSA, a classifier-agnostic feature selection architecture. FeSA evaluates feature robustness under simulated drift conditions and prioritizes temporally stable features over transiently discriminative ones. Experimental results show consistent superiority over genetic, evolutionary, and greedy selection techniques, underscoring that drift-aware feature engineering is as critical as adaptive classification for sustained ransomware detection.

Streaming analytics provides a complementary paradigm by treating ransomware detection as a continuous data stream problem rather than a static classification task. Ceschin et al. [8] model malware detection, including ransomware, over long-term Android datasets spanning nine years. By combining adaptive classifiers with drift detectors and periodically updating both feature extractors and models, the proposed pipeline achieves significant F1-score improvements. This work conclusively demonstrates that static ML pipelines are inadequate for real-world ransomware detection and reinforces the necessity of online and adaptive learning frameworks.

Beyond drift adaptation, several studies emphasize that behavioral feature quality directly impacts early detection capability. Bhagwat and Patil [77] analyze discriminative behavioral features derived from Windows API call traces and show that ransomware exhibits execution patterns distinct from other malware classes. Using these features, XGBoost and k-nearest neighbors achieve accuracies of 96.22% and 90%, respectively. The findings highlight that, in dynamic ransomware environments, feature discriminability often outweighs model complexity, particularly for early-stage detection.

The urgency of pre-encryption ransomware detection (PERD) has also motivated systematic synthesis efforts. Shaikh et al. [2] present a comprehensive literature review and taxonomy of PERD approaches, analyzing 30 studies published between 2018 and 2023. The review concludes that ML-based behavioral detection consistently outperforms traditional methods while identifying methodological weaknesses, evaluation inconsistencies, and gaps in early-detection benchmarking. The proposed taxonomy and research guidelines provide a structured foundation for future PERD research.

Several studies focus explicitly on minimizing detection latency before encryption commences. Davidian et al. [78] evaluate deep learning models for early ransomware detection using API call sequences enriched with execution context. By systematically analyzing window sizes, feature subsets, and model architectures, the study shows that convolutional neural networks and Long Short-Term Memory models with a context window of seven API calls achieve optimal performance. Importantly, incorporating operation results alongside API calls significantly improves precision, reinforcing the feasibility of real-time, pre-encryption deployment.

Hardware-assisted detection provides an even earlier vantage point for ransomware interruption. Mohan Anand et al. [23], previously discussed in the context of hardware-level behavior, are particularly relevant here for their timing advantage. Their HiPeR framework leverages hardware performance counters to detect ransomware within 3–4 seconds of execution—often before meaningful encryption begins. This capability positions hardware-level monitoring as a critical enabler of ultra-early ransomware detection, complementing software-level streaming approaches.

Hybrid early-detection strategies further enhance robustness by combining prior knowledge with runtime monitoring. Ayub et al. [6] propose RWArmor, a static-informed dynamic detection framework in which knowledge learned from static ML models guides real-time behavioral analysis. Evaluated on cryptographic Windows ransomware, RWArmor achieves 97.67% accuracy within 120 seconds and maintains strong performance even at 30 seconds of execution. This work demonstrates that informed feature fusion can significantly reduce detection latency without sacrificing accuracy.

To summarize hybrid static–dynamic architectures and drift-aware pre-encryption ransomware detection strategies, Table IV compares representative multi-tier frameworks, adaptation mechanisms, and early-detection objectives.

Critical synthesis. Across the dynamic-detection literature we observe a field converging on sequence models of API and system calls while simultaneously discovering the limits of that convergence. The near-perfect scores now routine for deep sequence models—often at or above 99%, occasionally 100% recall—are difficult to interpret without a shared benchmark: differences in sandbox configuration, family composition, and the treatment of benign software plausibly explain more of the variance than the model architecture does, and the scarcity of common datasets [42], [52] makes head-to-head comparison unsafe.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 06 , June - 2026

TABLE IV. HYBRID, DRIFT-AWARE, AND PRE-ENCRYPTION RANSOMWARE DETECTION FRAMEWORKS

Framework Type	Integrated Layers	Learning Paradigm	Adaptation / Drift Handling	Detection Stage	Key Contribution	Limitations	Representative Studies
Static–Dynamic Hybrid	Static binaries + runtime behavior	SVM	None	Pre-execution / Early runtime	Early feature fusion	Limited scalability	Hasan and Rahman [13]
Multi-Layer Hybrid	Static + dynamic + trap layer	ML	None	Pre-encryption	Early interception + recovery	Deployment overhead	Shaukat and Ribeiro [12]
Detection + Recovery	Network + honeypots + ML	XGBoost	None	Early detection + recovery	Operational resilience	Infrastructure cost	Pahuja et al. [71]
Network Propagation Detection	Network flows + ensemble fusion	ML ensembles	None	Pre-payload execution	Ransomworm containment	Network-only view	Almashhadani et al. [72]
Generic Hybrid ML	System calls + network + file behavior	Classical ML	None	Runtime	Reduced false positives	Feature tuning	Mumtaz et al. [73]
Hierarchical Deep Hybrid	Static IR + dynamic embeddings	LSTM-based DL	Confidence-triggered	Early runtime	Cross-platform scalability	Model complexity	Karbab et al. (SwiftR) [74]
Streaming Adaptive Detection	Dynamic behavior streams	Adaptive ML	Explicit drift detection	Continuous	Long-term robustness	Resource intensive	Ceschin et al. [8]
Drift-Aware Detection	Features + classifier layers	ML	Drift calibration & decision layers	Continuous	Stability under drift	Architectural complexity	Fernando and Komninos (FeSAD) [9]
Drift-Aware Feature Selection	Feature engineering stage	ML-agnostic	Feature robustness analysis	Pre-training	Drift-resilient features	Offline evaluation	Fernando and Komninos (FeSA) [76]
IoT Hybrid Framework	Memory forensics + ensembles	ML + NN	Concept drift handling	Runtime	Resource-aware hybridization	Dataset dependence	Abid et al. (ECMT) [75]
Hybrid Dataset Benchmark	Storage + runtime behavior	ML baseline	None	Evaluation support	Reproducible hybrid testing	Dataset-bound	Hirano et al. (RanSAP) [52]
Feature-Centric Early Detection	API traces	XGBoost, kNN	None	Pre-encryption	Feature discriminability	Platform-specific	Bhagwat and Patil [77]
PERD Taxonomy	Multi-layer review	—	—	Pre-encryption	Unified PERD taxonomy	Survey-only	Shaikh et al. [2]
Context-Aware PERD	API + execution context	DL (CNN, LSTM)	Window optimization	Pre-encryption	Latency minimization	Model tuning	Davidian et al. [78]
Hardware-Assisted Early Detection	HPCs	ML	None	Ultra-early	Detection in 3–4 s	Hardware dependence	Mohan Anand et al. (HiPeR) [23]
Static-Guided Dynamic PERD	Static ML + runtime monitoring	ML	Knowledge-guided	Early runtime	Reduced detection latency	Feature coupling	Ayub et al. (RWArmor) [6]

The shift toward low-level signals such as system calls, storage I/O, memory, and hardware performance counters [23], [24] buys evasion resistance at the cost of portability and deployment overhead, a trade-off acknowledged far more often than it is quantified. It is worth recalling that this trajectory builds on foundational behavioral and file-system defenses introduced a decade earlier—EldeRan [79], UNVEIL [80], ShieldFS [81], CryptoDrop [82], and PayBreak [83]—which established dynamic and decoy-based ransomware detection well before the current ML-centric wave.

Most consequential is that the dynamic paradigm is intrinsically reactive: by the time malicious behavior is observed, encryption may already be underway. The growing emphasis on pre-encryption windows, sliding-window early detection, and hardware-assisted termination [5], [6], [47] can be read as an implicit acknowledgment of this limitation, with detection latency—rather than accuracy—arguably the decisive metric for such methods. Tellingly, the studies that jointly deliver early detection, drift adaptation, and explainability [9], [22] remain a small minority, reinforcing the convergence requirement set out in Section 1.

6 PLATFORM-SPECIFIC RANSOMWARE DETECTION

Ransomware is not a monolithic threat but a platform-conditioned phenomenon: although its core objectives—data denial, extortion, and operational disruption—are consistent, the mechanisms used to achieve them are tightly coupled to system architecture, privilege models, file-system semantics, process isolation, and available interfaces, so detection strategies that succeed on one platform rarely generalize directly to others. Three factors drive this specificity. First, operating-system abstractions dictate how ransomware interacts with resources: Windows variants rely on Portable Executable (PE) structures, registry manipulation, and rich APIs; Android variants exploit permission models, sandboxing, and user-interface deception; and Linux variants operate at lower levels through system calls, runtime opcodes, and server-oriented execution. Second, the availability and granularity of telemetry differ across platforms, shaping which behavioral signals can be monitored reliably. Third, deployment constraints—resource limits in mobile and IoT settings, scalability demands in the cloud—directly affect which techniques are feasible.

Platform specificity also reshapes the threat model and attack lifecycle. On desktops, ransomware typically races from execution to encryption, prioritizing speed and persistence; on mobile devices, social engineering, permission abuse, and network communication dominate; and in IoT and cyber–physical systems (CPS), attacks often target availability rather than confidentiality, exploiting lateral propagation and control-channel disruption. Effective detectors must therefore be not only behavior-aware but context-aware, accounting for platform-dependent execution semantics and attacker incentives.

Platform-centric detection thus embodies a trade-off between accuracy and generalization: tight coupling to OS-specific artifacts yields precise detection but limited portability, whereas platform-agnostic approaches generalize at

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 06 , June - 2026

the cost of fine-grained behavioral signals—a tension that is especially consequential in enterprise and critical-infrastructure environments where heterogeneous platforms coexist. The following subsections review detection research organized by target platform—Windows, Android, Linux, IoT, and heterogeneous systems—highlighting how platform characteristics shape feature selection, modeling strategy, and performance while exposing the generalization limits that motivate the cross-platform and hybrid approaches discussed later.

6.1 Windows-Based Ransomware Detection

Microsoft Windows remains the dominant target platform for ransomware attacks due to its widespread adoption across enterprise, government, and personal computing environments. As a result, many detection frameworks are tightly coupled with Windows-specific artifacts, including Portable Executable (PE) binaries, Windows API calls, registry operations, file-system behavior, and execution logs. These approaches often achieve high detection accuracy but are inherently platform-dependent.

Early Windows-focused detection emphasizes rapid pre-execution analysis. Khammas [15] proposes a static ransomware detection framework based on Random Forest classifiers trained on raw byte-level patterns, deliberately avoiding disassembly to reduce analysis latency. Through frequent pattern mining and Gain Ratio–based feature selection, the feature space is reduced to 1000 discriminative byte sequences, achieving 97.74% detection accuracy. Although purely static, the approach is well suited for time-critical Windows environments where blocking ransomware prior to execution is essential.

Given the richness of Windows telemetry, behavioral feature selection plays a central role in improving detection reliability. Malik et al. [84] conduct a systematic analysis of Windows execution logs, identifying behavioral features that most strongly influence ransomware classification. Their findings demonstrate that careful feature selection yields substantial accuracy improvements across multiple machine learning (ML) classifiers, reinforcing that feature relevance often outweighs model complexity, particularly for early-stage detection.

Dynamic behavioral analysis further enhances Windows ransomware detection. Jaya and Ab Razak [85] investigate runtime detection using the TON_IoT dataset in a controlled Windows 7 environment. By extracting 77 behavioral features reflecting system changes before and after infection, the authors evaluate multiple classifiers and report 99.74% accuracy with an Area Under the Curve (AUC) of 0.997 using Random Forest and J48. This study confirms the effectiveness of runtime ML-based detection for high-confidence ransomware identification on Windows platforms.

Hybrid static–dynamic frameworks are particularly prevalent in Windows environments due to the availability of complementary telemetry. Hasan and Rahman [13] introduce RansHunt, which integrates static binary features with dynamic behavioral indicators using Support Vector Machines (SVMs). Evaluated on multiple ransomware families, including WannaCry, RansHunt demonstrates improved robustness against sandbox-aware and intrusion detection system (IDS)–aware ransomware, illustrating the benefits of hybrid feature fusion.

Layered Windows-specific defense architectures further extend this paradigm. Shaukat and Ribeiro [12] propose RansomWall, a multi-layer ransomware defense framework combining static inspection, dynamic monitoring, and ML-based classification. A distinctive contribution of RansomWall is its trap layer, which enables early detection while backing up suspicious file modifications to prevent irreversible data loss. Tested on 574 ransomware samples from 12 families, the framework achieves a 98.25% detection rate with near-zero false positives, demonstrating suitability for real-world Windows deployments.

Early-stage detection under partial observability is addressed by Sukul et al. [5], who propose an augmented bootstrapping technique based on timestamped API call N-grams weighted using Term Frequency–Inverse Document Frequency (TF–IDF). The framework achieves high accuracy even with incomplete early execution traces, directly targeting pre-encryption detection—a critical requirement in Windows environments where encryption progresses rapidly.

Windows API semantics remain a dominant detection surface. Mohan Anand et al. [14] perform a large-scale analysis of API call importance across 46 ransomware families, identifying 135 consistently discriminative APIs. Ensemble classifiers trained on these features achieve 96.15% accuracy, providing a lightweight and interpretable detection strategy suitable for endpoint security solutions.

Recent Windows-focused research increasingly emphasizes explainability. Tan and Abdulnabi [49] propose TriFRIM, a triple-filter interpretable ransomware detection model integrating signature-based, network-based, and behavioral features. Although largely conceptual, the framework highlights the need for human-centered decision support in Windows security operations.

Explainable deep learning approaches further advance this direction. Gulmez et al. [20] introduce XRan, an explainable dynamic ransomware detection framework modeling Windows API calls, DLL activity, and mutex operations using CNN-based sequence learning combined with SHAP and LIME explanations. Achieving up to 99.4% true positive rate, XRan demonstrates that high detection accuracy and interpretability are not mutually exclusive.

6.2 Android and Mobile Ransomware Detection

Android ransomware detection presents distinct challenges due to application sandboxing, permission-based security models, heterogeneous hardware, and limited system visibility. Mobile ransomware frequently exploits network communication, deceptive user interfaces, and permission abuse, necessitating platform-aware detection strategies.

Network-centric detection has emerged as a practical solution where host-level instrumentation is restricted. Mahinderjit Singh et al. [86] propose a network-traffic–based Android ransomware detection framework extracting 84 discriminative flow features. Using k-means clustering for feature selection and Random Forest classification, the framework achieves 95.18% accuracy, providing a complementary detection layer suitable for resource-constrained mobile deployments.

Dynamic system behavior remains a strong indicator of Android ransomware activity. Abdullah et al. [87] investigate system-call–based dynamic analysis, reporting 98.31% accuracy using Random Forest with low false-positive rates. This work demonstrates that runtime system-call traces effectively characterize ransomware behavior on mobile platforms.

Hybrid static–dynamic detection further enhances robustness. Gharib and Ghorbani [34] introduce DNA-Droid, a real-time framework integrating static analysis with dynamic sandbox execution. Behavioral similarity is modeled using sequence alignment techniques, enabling detection of

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 06 , June - 2026

previously unseen ransomware families with a false-negative rate below 1.5%.

Adaptive deep learning addresses the rapid evolution of mobile ransomware. Sharmeen et al. [88] propose a semi-supervised framework that learns latent behavioral patterns from unlabeled samples and integrates them with supervised classifiers, outperforming conventional methods under limited labeling scenarios.

Efficiency remains critical for mobile platforms. Bau et al. [89] show that Random Forest consistently outperforms ANN and CNN models in both accuracy and computational efficiency for multiclass Android malware detection, reinforcing the continued relevance of classical ML under resource constraints.

Hybrid representations incorporating textual semantics are also explored. Arputha Rathina et al. [70] integrate static features, dynamic behavior, and threat-text extraction from ransom notes, achieving up to 91% accuracy with low false negatives.

Adversarial robustness is a growing concern. Cara et al. [36] demonstrate that API-frequency–based Android detectors can be evaded through semantic-preserving API injection, exposing vulnerabilities in behavior-based models and motivating adversarial-aware detection strategies.

6.3 Linux-Based Ransomware Detection

Although historically less targeted, Linux systems have become increasingly attractive ransomware targets in servers, cloud infrastructures, containers, and IoT gateways. Research in this domain emphasizes low-level, platform-agnostic behavioral features.

Alonso et al. [90] investigate runtime opcode tracing for Linux malware detection, capturing executed instruction sequences during execution. Evaluating multiple classifiers, the study reports over 90% accuracy using Random Forest, demonstrating the discriminative power of opcode-level behavioral traces.

Extending this work, Alonso et al. [26] compare static binary inspection with dynamic opcode tracing using both classical ML and an LSTM-based model. Results show complementary strengths, with dynamic Random Forest achieving ≥90% accuracy and static LSTM reaching 98%. Although not exclusively ransomware-focused, the study is highly relevant due to its emphasis on behavioral learning in non-Windows environments.

6.4 IoT and Cyber–Physical Systems (CPS)

The proliferation of IoT and CPS has expanded the ransomware attack surface to safety- and availability-critical environments. These systems are characterized by heterogeneity, constrained resources, and limited visibility. Maghanaki et al. [91] address dataset scarcity by introducing a multi-class IoT malware dataset with a dedicated ransomware class, comprising 10,000 samples and 171 behavioral features collected via sandbox execution.bAbid et al. [75] propose ECMT, a memory-centric detection framework integrating in-memory forensic attributes with ensemble learning, achieving up to 96% accuracy while addressing scalability and concept drift.

Alissa et al. [32] introduce an optimization-driven IoT ransomware detection framework combining Dwarf Mongoose Optimization with an Extreme Learning Machine to reduce feature dimensionality under strict resource constraints.

Network-level detection addresses lateral propagation. Almashhadani et al. [72] propose MFMCNS, a multi-feature, multi-classifier system detecting ransomworm activity during propagation, enabling early containment in IoT and CPS deployments.

6.5 Cross-Platform and Generic Ransomware Detection Frameworks

Modern ransomware campaigns increasingly span heterogeneous environments, motivating cross-platform detection frameworks that focus on behavioral semantics rather than OS-specific artifacts.

Saranya et al. [92] propose a context-aware malware scoring framework integrating structured features with unstructured textual data using BERT embeddings and ensemble learning. Although ransomware is not treated as a standalone class, the approach supports cross-platform generalization and risk scoring.

Jain and Nihalani [93] provide a comprehensive review of ML-based botnet detection, offering insights into the infrastructure supporting ransomware campaigns. Botnet-level indicators serve as early warning signals for pre-compromise ransomware detection.

To consolidate platform-specific ransomware detection strategies and highlight their strengths and generalization limitations, Table V summarizes representative approaches across major execution environments.

7 ADVANCED AND EMERGING RANSOMWARE DETECTION TECHNIQUES

The rapid evolution of ransomware has exposed fundamental limits in detection paradigms built on fixed feature sets, shallow classifiers, and static threat assumptions. Modern families increasingly exhibit adaptive execution, context-aware triggering, multi-stage lifecycles, and deliberate evasion of monitoring, rendering traditional static and dynamic pipelines insufficient in isolation and motivating a shift from feature-centric detection toward representation-centric, resilient, and adaptive models. Four challenges drive this shift. First, ransomware behavior is high-dimensional and multimodal—spanning code structure, runtime execution, system interaction, memory state, and microarchitectural effects—so capturing it requires learned representations rather than handcrafted features. Second, malicious intent is temporally and contextually dependent, emerging only through long-range execution patterns, delayed triggers, or coordinated cross-process actions. Third, concept drift and adversarial adaptation continuously degrade static models, demanding self-adapting, robust learning. Finally, deployment constraints—latency, scalability, privacy, and trust—require architectures that are operationally viable, not merely accurate.

In response, recent research has converged on paradigms emphasizing deep representation learning, hardware-assisted monitoring, graph-based relational reasoning, privacy-preserving collaboration, and evasion-aware modeling. These approaches seek intrinsic behavioral signatures that ransomware cannot conceal without losing functionality, enabling early-stage and zero-day detection, and they increasingly blur the boundaries between static and dynamic analysis, software and hardware telemetry, and local and distributed intelligence—reflecting a holistic view of ransomware as a cross-layer cyber–physical threat.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 06 , June - 2026

TABLE V. PLATFORM-SPECIFIC RANSOMWARE DETECTION: CHARACTERISTICS, FEATURES, AND LIMITATIONS

Target Platform	Dominant Ransomware Traits	Primary Feature Sources	Detection Paradigm	Key Strengths	Generalization Limitations	Representative Studies
Windows (Desktop / Enterprise)	Rapid encryption, registry persistence, PE-based execution	Raw bytes, PE metadata, Windows APIs, registry, execution logs	Static dynamic ML hybrid, XAI	High accuracy, rich telemetry, early detection	OS-version dependence, limited portability	Khammas [15]; Malik et al. [84]; Jaya and Ab Razak [85]; Hasan and Rahman [13]; Shaukat and Ribeiro [12]; Sukul et al. [5]; Mohan Anand et al. [14]; Tan and Abdulnabi [49]; Gulmez et al. [20]
Android / Mobile	Permission abuse, UI deception, network reliance	Permissions, system calls, network flows, dynamic traces	Network-based ML, dynamic ML, hybrid, semi-supervised DL	Resource efficiency, adaptability	Limited visibility, adversarial evasion	Mahinderjit Singh et al. [86]; Abdullah et al. [87]; Gharib and Ghorbani [34]; Sharmeen et al. [88]; Bau et al. [89]; Arputha Rathina et al. [70]; Cara et al. [36]
Linux (Servers / Cloud)	Opcode-level behavior, low-level execution, server targeting	Runtime opcodes, binary inspection	Dynamic ML, LSTM-based DL	Platform-agnostic semantics	Runtime overhead, limited datasets	Alonso et al. [90]; Alonso et al. [26]
IoT / CPS	Availability disruption, lateral propagation, constrained execution	Behavioral features, memory forensics, network flows	Ensemble ML, optimization-driven ML	Lightweight detection, propagation awareness	Dataset scarcity, safety constraints	Maghanaki et al. [91]; Abid et al. [75]; Alissa et al. [32]; Almashhadani et al. [72]
Cross-Platform / Generic	Multi-stage campaigns, ecosystem-level behavior	Behavioral semantics, text + structured data	Context-aware ML, NLP-based DL	Improved generalization	Label ambiguity, reduced precision	Saranya et al. [92]; Jain and Nihalani [93]

This section surveys these advanced and emerging techniques, organized by their underlying detection philosophy—representation learning, hardware-level analysis, relational modeling, decentralized trust, and adversarial robustness—highlighting both their theoretical contributions and their practical implications for next-generation ransomware defense.

7.1 Deep Learning and Representation Learning

Recent ransomware detection research has increasingly transitioned from handcrafted features and shallow classifiers toward deep representation learning. This shift is motivated by the need to capture complex temporal dependencies, semantic context, and low-level behavioral patterns that are difficult to encode explicitly. Deep learning models—particularly convolutional neural networks (CNNs), recurrent neural networks (RNNs), Transformers, and attention-based hybrids—enable automated feature extraction across heterogeneous data modalities, including API call sequences, opcode streams, hardware performance counters, memory traces, and behavioral logs.

7.1.1 Behavior-to-Representation Transformations

A prominent trend in advanced ransomware detection is the transformation of behavioral artifacts into representations that are more amenable to deep learning. Kesharwani et al. [16] introduce a behavior-to-vision transformation for Android ransomware detection, converting JSON-based sandbox reports into RGB images. This representation enables the use of CNNs and Vision Transformers (ViTs), with the ViT model achieving 99.78% accuracy and outperforming CNN-based baselines. By spatially encoding behavioral semantics, the approach captures complex correlations and temporal relationships that are difficult to exploit using conventional sequence models.

Similarly, Ganfure et al. [24] propose DeepWare, which converts hardware performance counter (HPC) distributions into image-like representations and applies CNNs for classification. Using execution snapshots as short as 100 ms, DeepWare achieves 98.6% recall with near-zero false positives and demonstrates strong generalization to previously unseen ransomware families. These works collectively highlight the effectiveness of cross-domain encoding, where non-visual behavioral signals are reinterpreted as visual patterns for robust deep feature extraction.

7.1.2 Sequence- and Context-Aware Deep Architectures

Sequence modeling remains central to dynamic ransomware detection, particularly for API calls, system calls, and I/O traces. Chen et al. [19] propose RANsomCheck, a CNN–Transformer hybrid architecture that combines local pattern extraction with long-range dependency modeling. A novel quadruple-based semantic encoding enriches contextual representation, enabling the model to achieve 99.94% accuracy and 100% recall. The integration of CNNs with Transformers reflects a broader architectural trend toward balancing efficiency with expressive power.

In a complementary direction, Jahromi et al. [4] introduce an enhanced stacked Long Short-Term Memory (LSTM) architecture with unsupervised layer-wise pretraining to eliminate random initialization. The proposed model exhibits stable convergence and strong robustness across multiple datasets, achieving 99.1% accuracy with high Matthews Correlation Coefficient (MCC) and Area Under the Curve (AUC) scores. This work underscores the importance of training stability and initialization strategies in time-critical ransomware detection systems.

7.1.3 Semantic Embeddings and Language-Inspired Models

Inspired by natural language processing, several studies treat ransomware behavior as an "execution language." Alvi et al. [46] propose RGV2, which leverages FastText embeddings to encode low-level file I/O patterns extracted from Event Tracing for Windows (ETW) logs. These semantic embeddings capture contextual relationships between events and are classified using deep neural networks, achieving 99.64% accuracy with low false-positive rates. Unlike purely syntactic sequence models, semantic embeddings enhance generalization to unseen ransomware variants, particularly during encryption phases.

Transformer-based language models further extend this paradigm by enabling context-aware representation learning across long execution traces. Such models are increasingly

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 06 , June - 2026

favored for handling evolving ransomware behaviors, where local event patterns alone are insufficient to characterize malicious intent.

7.1.4 Adaptive and Semi-Supervised Deep Learning

To address the rapid evolution of ransomware and the scarcity of labeled data, adaptive and semi-supervised deep learning approaches have gained attention. Sharmeen et al. [88] propose a deep adaptive framework that learns latent behavioral patterns from unlabeled ransomware samples and integrates them with supervised classifiers. The framework demonstrates superior performance compared to fully supervised baselines and is particularly effective for zero-day and emerging ransomware families.

By reducing dependence on exhaustive labeling, adaptive learning paradigms improve resilience against concept drift, which remains a critical limitation of static deep learning pipelines in long-term ransomware defense.

7.1.5 Hardware-Assisted Deep Learning

Hardware-level signals offer a powerful and evasion-resistant feature source for ransomware detection. While HiPeR [23] primarily employs classical machine learning over hardware performance counters, it establishes the foundation for deep learning–based ultra-early detection, achieving detection within 3–4 seconds of execution. When combined with representation learning techniques—such as behavioral imaging in DeepWare [24]—hardware-assisted deep learning emerges as a promising direction for pre-encryption and zero-day ransomware detection.

7.2 Hardware, Performance Counter, and Side-Channel–Based Detection

As ransomware increasingly incorporates obfuscation, sandbox evasion, and anti-instrumentation techniques, software-level monitoring mechanisms—such as API hooking and system-call tracing—have become more susceptible to bypass. To mitigate these limitations, recent research explores hardware-assisted ransomware detection using performance monitoring counters and other microarchitectural side channels as low-level, operating-system–independent behavioral signals.

Mohan Anand et al. [23] introduce HiPeR, one of the earliest frameworks demonstrating that hardware performance counters can be used for ultra-early ransomware detection. By monitoring a compact set of CPU-level events, HiPeR detects ransomware within 3–4 seconds of execution, enabling termination before encryption completes. Because this approach operates independently of OS-level hooks, it is resilient to system-call obfuscation and API tampering.

Building on this foundation, Ganfure et al. [24] extend hardware-based detection using deep learning, transforming short execution snapshots into image-like representations and applying CNNs for classification. The resulting framework achieves high recall with minimal false positives and strong generalization to unseen ransomware families.

To address scalability and deployment feasibility, Woralert et al. [55] propose a LightGBM-based detection framework optimized for performance counter data. The framework achieves 99.95% binary accuracy and supports rapid adaptation to new ransomware variants via transfer learning, making it suitable for cloud and high-performance computing environments.

7.3 Graph-Based Modeling, Knowledge Graphs, and Clustering

Traditional feature-vector–based ransomware detection often fails to capture structural relationships and multi-entity interactions inherent in complex ransomware behavior. Graph-based modeling addresses this limitation by enabling relational reasoning and improved generalization.

Satpathy and Swain propose Graph-Contrast Ransomware Detection (GCRD) [17], which models PE file features as graphs and applies Graph Neural Networks (GNNs) with contrastive learning. The final Transformer-based classifier achieves 99.1% accuracy with low inference latency, demonstrating suitability for real-time enterprise deployment.

Rosli et al. [57] explore behavioral graph construction using dynamic file-system activities stored in a Neo4j graph database, enabling visualization and forensic analysis of ransomware execution paths. Although evaluated on a limited dataset, the study highlights the interpretability benefits of graph-based abstraction.

To address unseen ransomware families, Wang et al. [94] propose a transductive zero-shot learning framework built on a malware knowledge graph. By embedding semantic relationships between ransomware families, the approach achieves an F1-score of 93.5% under the generalized zero-shot learning protocol, substantially outperforming CNN baselines.

Nguyen et al. [50] further extend graph-inspired reasoning to cloud environments by combining static and dynamic features with clustering and data synthesis using Conditional Tabular GANs.

7.4 Blockchain, Privacy-Preserving, and Federated Detection Frameworks

As ransomware detection increasingly relies on distributed data sources, concerns regarding data integrity, trust, and privacy have motivated blockchain-enabled and privacy-preserving detection frameworks. Danya et al. [95] propose a blockchain-assisted ransomware detection system leveraging the Solana blockchain for decentralized trust management. While still exploratory, this work highlights blockchain's potential for secure collaboration and auditability in distributed detection ecosystems.

7.5 Adversarial Machine Learning and Evasion-Aware Detection

Despite high reported accuracy, many ransomware detectors remain vulnerable to adversarial evasion. Cara et al. [36] demonstrate that Android malware detectors relying on API frequency features can be evaded through semantic-preserving API injection. De Gaspari et al. [10] further show that cooperative multi-process attacks can reduce detection accuracy to zero, exposing fundamental limitations of single-process behavioral modeling. Their earlier work [11] confirms that such evasion strategies can defeat state-of-the-art behavioral classifiers.

To synthesize advanced and emerging ransomware detection paradigms beyond traditional static and dynamic pipelines, Table VI summarizes representative techniques according to their underlying detection philosophy, feature abstraction level, and resilience properties.

8 DATASETS, FEATURE ENGINEERING, AND EVALUATION PRACTICES

Ransomware detection systems are only as reliable as the data, features, and evaluation protocols on which they are built. Because ransomware exhibits time-sensitive execution, family-specific encryption logic, adaptive evasion, and rapid concept drift, dataset construction and experimental validation are especially difficult, and performance claims resting on high accuracy alone—without attention to data provenance,

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 06 , June - 2026

TABLE VI. ADVANCED AND EMERGING RANSOMWARE DETECTION TECHNIQUES: PARADIGMS, CAPABILITIES, AND LIMITATIONS

Detection Paradigm	Core Theoretical Motivation	Primary Signals / Representations	Key Advantages	Primary Limitations	Representative Studies
Deep Representation Learning	Capture high-dimensional, multimodal ransomware behavior beyond handcrafted features	API sequences, opcodes, I/O traces, memory artifacts	Automated feature discovery, strong generalization	Computational overhead, explainability challenges	Kesharwani et al. [16]; Chen et al. [19]; Jahromi et al. [4]
Behavior-to-Representation Transformation	Reinterpret behavioral data into learnable visual or spatial domains	RGB images from sandbox reports or HPCs	Robust feature abstraction, high detection accuracy	Transformation cost, deployment complexity	Kesharwani et al. [16]; Ganfure et al. [24]
Sequence- and Context-Aware Modeling	Model long-range temporal dependencies and delayed execution patterns	API call sequences, system-call streams	Early-stage detection, temporal sensitivity	Sensitive to windowing and execution truncation	Chen et al. [19]; Jahromi et al. [4]
Semantic Embeddings and Language-Inspired Models	Treat ransomware execution as a contextual "behavioral language"	ETW-based file I/O embeddings, event semantics	Improved generalization to unseen variants	Training complexity, interpretability	Alvi et al. [46]
Adaptive and Semi-Supervised Learning	Address concept drift and scarcity of labeled ransomware samples	Latent behavioral representations	Zero-day resilience, reduced labeling dependence	Stability and convergence challenges	Sharmeen et al. [88]
Hardware-Assisted Deep Detection	Exploit OS-independent microarchitectural signals for ultra-early detection	Hardware performance counters (HPCs)	Evasion resistance, pre-encryption detection	Hardware dependency, limited interpretability	Mohan Anand et al. [23]; Ganfure et al. [24]
Hardware-Level ML (Non-DL)	Enable lightweight and scalable detection using low-level execution traces	CPU-level events, performance counters	Low latency, cloud suitability	Architecture specificity	Woralert et al. [55]
Graph-Based and Relational Modeling	Capture structural and relational dependencies across entities	PE graphs, behavioral graphs, knowledge graphs	Cross-family generalization, interpretability	Graph construction overhead	Satpathy and Swain [17]; Rosli et al. [57]; Wang et al. [94]; Nguyen et al. [50]
Blockchain and Privacy-Preserving Detection	Ensure trust, integrity, and collaboration across distributed detectors	Decentralized ledgers, secure metadata	Tamper resistance, auditability	Immature tooling, scalability	Danya et al. [95]
Adversarial and Evasion-Aware Detection	Model attacker adaptation and adversarial manipulation	API injection patterns, multi-process behavior	Exposure of detector weaknesses	Increased system complexity	Cara et al. [36]; De Gaspari et al. [10]; De Gaspari et al. [11]

feature stability, and evaluation realism—often fail to translate into operational effectiveness. A central problem is the mismatch between laboratory datasets and real-world behavior: many datasets are limited in diversity, outdated, class-imbalanced, or built in synthetic environments that miss realistic encryption workflows and evasion. Moreover, because detection is prevention-oriented, early-stage latency, false-positive cost, and robustness under partial observability often matter more than post-hoc classification accuracy—constraints poorly served by standard malware benchmarks.

Feature engineering adds a second layer of complexity. Although deep learning has reduced reliance on handcrafted features, ransomware detection still depends on behavioral signals—system calls, API usage, file-system activity, and storage-access patterns—that are highly sensitive to noise, evasion, and execution context. Feature relevance must therefore be judged not only by discriminative power but by temporal stability, resilience to adversarial manipulation, and robustness under drift; strategies that optimize short-term accuracy without these properties degrade rapidly in deployment.

Evaluation methodology is equally decisive. Static train–test splits, balanced datasets, and aggregate accuracy obscure weaknesses in family-wise generalization, zero-day performance, and early detection; ransomware-specific evaluation instead demands time-aware metrics, drift-sensitive validation, and deployment-oriented assumptions, particularly when decisions must precede irreversible encryption. Motivated by these challenges, recent work increasingly treats dataset transparency, behavior-aware feature engineering, and rigorous drift-aware evaluation as first-class contributions. This section surveys key efforts in dataset construction, feature selection, and evaluation methodology that underpin the reliability, reproducibility, and real-world relevance of ransomware detection.

8.1 Dataset Construction and Availability

One of the persistent bottlenecks in ransomware research is the lack of public, well-curated, and behaviorally grounded datasets. Maghanaki et al. [91] address this gap by introducing a multi-class Internet of Things (IoT) malware dataset that explicitly incorporates ransomware as a dedicated class. Unlike monolithic datasets that conflate malware behaviors, the proposed dataset isolates behavior-specific characteristics, thereby improving discriminative learning. The dataset comprises 10,000 samples and 171 behavioral features collected via sandbox execution on the AnyRun platform and is balanced across benign, suspicious, and malicious classes. Although no detection model is proposed, the dataset provides a foundational benchmark for evaluating ransomware detection in resource-constrained and heterogeneous IoT environments.

Complementing this effort, Herrera Silva and Hernández-Alvarez [42] present a public dynamic feature dataset derived from sandbox execution of encryptor and locker ransomware. The dataset includes 50 carefully selected runtime features spanning multiple platforms and demonstrates that models trained on well-curated dynamic features can achieve detection accuracies exceeding 99%. This contribution is particularly important for reproducibility and comparative evaluation, which remain weak points in ransomware research.

At a lower system level, Hirano et al. [52] introduce RanSAP, an open dataset capturing ransomware storage access patterns using a live-forensic hypervisor. The dataset includes traces from multiple ransomware variants, benign applications, operating systems, and storage configurations, including full-disk encryption. By providing both the dataset and a baseline machine learning (ML) pipeline, RanSAP enables controlled evaluation of storage-centric ransomware detection, a behavior that is tightly coupled with encryption activity.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 06 , June - 2026

TABLE VII. DATASETS, FEATURE ENGINEERING, AND EVALUATION PRACTICES IN RANSOMWARE DETECTION

Aspect	Theoretical Motivation	Key Characteristics	Strengths	Limitations / Open Issues	Representative Studies
Public Ransomware Datasets	Enable reproducibility and comparative evaluation	Sandbox-executed behavioral traces; multi-class labeling	Improves benchmarking and transparency	Limited diversity; controlled environments	Herrera Silva and Hernández-Alvarez [42]
IoT-Centric Malware Datasets	Address dataset scarcity in resource-constrained platforms	10,000 samples; 171 behavioral features; ransomware as explicit class	Supports heterogeneous and IoT-focused evaluation	No integrated detection pipeline	Maghanaki et al. [91]
Storage-Level Behavioral Datasets	Capture encryption-coupled ransomware behavior	Hypervisor-based storage access traces; OS diversity	Strong coupling to encryption activity; low evasion	Limited to storage-centric analysis	Hirano et al. [52]
Behavioral Feature Engineering	Improve early detection and robustness	API calls, system calls, execution logs	High discriminative power; interpretable	Sensitive to noise and evasion	Malik et al. [84]
Feature Refinement and Sanitization	Remove adversarial and noisy behavioral artifacts	Refined system-call traces; EmRmR selection	Reduced false positives; improved early detection	Requires careful preprocessing	Ahmed et al. [96]
Drift-Aware Feature Selection	Sustain detection under evolving ransomware behavior	Drift calibration and decision layers	Long-term stability; reduced retraining	Increased system complexity	Fernando and Komninos [9]
Conventional Evaluation Protocols	Measure classification performance	Static train–test splits; accuracy, F1-score	Simple and widely adopted	Masks early-detection failure; weak realism	Widely used across studies
Time-Aware and Pre-Encryption Evaluation	Reflect real-world prevention requirements	Early execution windows; latency-aware metrics	Operational relevance; realistic assessment	Inconsistent adoption	Herrera Silva and Hernández-Alvarez [42]; Ahmed et al. [96]
Drift-Aware Evaluation Methodologies	Assess long-term deployment viability	Temporal validation; drift-sensitive testing	Robustness under evolving threats	Requires long-term datasets	Fernando and Komninos [9]
Explainability-Guided Evaluation	Improve trust and feature stability	Feature-importance analysis (e.g., SHAP)	Enhances interpretability and robustness	Added computational overhead	Discussed in [9]

8.2 Feature Engineering and Selection

Beyond data availability, feature quality has repeatedly been shown to outweigh model complexity in ransomware detection. Ahmed et al. [96] propose a system-call–refinement-based enhanced Minimum Redundancy Maximum Relevance (EmRmR) feature selection method for early ransomware detection. By filtering out noisy and intentionally injected system calls used for evasion, the refined feature set significantly improves early detection accuracy while reducing false positives. This work highlights that raw behavioral traces must be carefully sanitized before learning.

Malik et al. [84] further emphasize this observation through a critical analysis of behavioral feature selection using Windows execution logs. Their results show that identifying a small subset of highly discriminative behavioral features yields larger performance gains than switching between classifiers. The study reinforces that feature relevance, rather than model sophistication, is often the limiting factor in practical ransomware detection.

Addressing long-term deployment challenges, Fernando and Komninos introduce FeSAD [9], a concept-drift-aware feature selection and detection framework. By integrating drift calibration and drift decision layers, FeSAD maintains reliable detection performance under evolving ransomware behaviors without frequent retraining. This work is foundational for production-grade ransomware detection, where static feature sets rapidly become obsolete.

8.3 Evaluation Methodologies and Metrics

Evaluation practices in ransomware research vary widely, often limiting comparability across studies. While many works report high accuracy or F1-scores, fewer explicitly address early detection latency, false-positive cost, or robustness under partial observability. Studies such as [42] and [96] demonstrate that pre-encryption evaluation windows and time-aware metrics are critical for assessing real-world effectiveness.

Moreover, ransomware-specific challenges—including class imbalance, family-wise generalization, and concept drift—necessitate evaluation beyond static train–test splits. Drift-aware evaluations, as adopted in [9], and feature-importance analyses using explainability tools (e.g., SHAP-based studies discussed earlier) are increasingly essential for trustworthy and deployment-ready assessment.

Summary:
Collectively, these studies demonstrate that progress in ransomware detection is tightly coupled with advances in dataset design, feature engineering discipline, and evaluation rigor. Public datasets, refined behavioral features, and drift-aware evaluation protocols are indispensable for transitioning ransomware detection research from benchmark-driven experimentation to operationally reliable defense systems.

On the (in)comparability of reported accuracies. A recurring theme of this survey is that the headline accuracies reported across the literature—commonly between 97% and 100%—should not be read as a ranking of techniques. Because studies differ in their datasets, family composition, class balance, sandboxing, observation windows, and the proportion of benign software, a model reporting 99.9% on one corpus and another reporting 95% on a different corpus are simply not measured on the same axis; the gap may reflect data difficulty rather than detector quality. Two consequences deserve emphasis. First, aggregate accuracy and F1 obscure precisely the quantities that govern deployment: pre-encryption detection latency, the false-positive rate at realistic base rates, and family-wise or temporal generalization. In an enterprise generating millions of benign events, even a 0.1% false-positive rate can translate into an unmanageable alert volume—a base-rate effect that high accuracy conceals. Second, very few studies release code and data together, so reported numbers are rarely independently reproducible. Fair progress in this field therefore depends less on new architectures than on shared benchmarks, time-aware and cost-sensitive metrics, and drift-aware evaluation protocols; until these are standard, cross-study accuracy comparisons should be treated as indicative at best.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 06 , June - 2026

To consolidate methodological advances and limitations in ransomware research, Table VII summarizes representative datasets, feature engineering strategies, and evaluation practices, highlighting their theoretical contributions and practical constraints.

9 ADDITIONAL SURVEYS AND HIGH-LEVEL PERSPECTIVES

As ransomware detection research has rapidly expanded, several surveys have emerged to consolidate knowledge, identify trends, and outline open challenges. Among these, Urooj et al. [1] provide a comprehensive survey of ransomware detection using dynamic analysis and machine learning across Windows, Android, IoT, and cloud platforms.

The survey systematically reviews datasets, feature extraction strategies, and machine learning (ML) and deep learning (DL) techniques published between 2019 and 2021, highlighting dynamic analysis as a unifying detection paradigm. A key contribution of this work is its identification of persistent research gaps, including limited availability of public datasets, insufficient emphasis on real-time and pre-encryption detection, and lack of cross-platform generalization. The authors also outline future research directions, making the survey a foundational reference for studies targeting behavior-based ransomware detection in evolving and heterogeneous environments.

Beyond this single reference point, several broader surveys frame the field. Razaulla et al. [97] trace the evolution, taxonomy, and defenses of ransomware across more than 150 works, whereas Or-Meir et al. [98] provide an authoritative account of dynamic malware analysis that underpins much of the behavioral tradition surveyed here. More recently, attention has turned toward purpose-built behavior-based benchmarks [99]. To position the present review against these efforts, Table VIII contrasts representative surveys along the dimensions that motivated our taxonomy; the comparison highlights that, unlike prior surveys, this work treats pre-encryption detection, concept drift, explainability, and adversarial robustness as first-class concerns and is, to our knowledge, among the first ML-focused ransomware surveys organized through an explicit PRISMA protocol with stated research questions.

10 CONCLUSION AND FUTURE RESEARCH DIRECTIONS

Answers to the research questions. Returning to the questions posed in Section 1, this review supports the following answers.

RQ1. Detection is organized along four dimensions—target platform, analysis level, learning paradigm, and detection stage (Figure 2)—with API- and system-call features and sequence models dominating, and Windows remaining the most heavily studied platform (Sections 2–5).

RQ2. Genuine pre-encryption and early-stage detection remains the exception rather than the rule; most methods still classify behavior after encryption has begun, although static-informed and sliding-window approaches are steadily narrowing this gap (Sections 4 and 6).

RQ3. Robustness to concept drift and adversarial evasion is seldom evaluated rigorously; few studies test temporal generalization or adaptive attackers, leaving many resilience claims largely unverified (Sections 4, 6, and 7).

RQ4. Reported results rest on heterogeneous datasets, feature pipelines, and evaluation protocols, and code and data are rarely released together, so cross-study accuracy figures are not directly comparable and reproducibility remains limited (Section 7).

RQ5. The principal open challenges—pre-encryption latency, drift-aware adaptation, explainability, adversarial robustness, cross-platform generalization, and standardized benchmarking—define the research agenda detailed below (Sections 7–9).

This review presented a comprehensive and structured analysis of machine learning–based ransomware detection research, spanning static analysis, dynamic behavior modeling, hybrid frameworks, platform-specific solutions, and advanced emerging techniques. By organizing the literature along orthogonal dimensions—including analysis level, learning paradigm, detection stage, and deployment context—the review clarifies how ransomware detection has evolved from signature-driven defenses toward behavior-centric, adaptive, and intelligence-driven security systems.

The surveyed studies collectively demonstrate that ransomware detection is no longer a single-layer classification problem but a multi-faceted challenge involving early detection under partial observability, resilience to concept drift, robustness against adversarial evasion, and practical deployability across heterogeneous platforms. Static analysis remains valuable for early-stage screening and cross-platform coverage, while dynamic behavior modeling—particularly sequence-based learning over API calls, system calls, and file-system activity—dominates high-accuracy detection. Hybrid static–dynamic architectures and multi-tier defense frameworks further improve robustness by integrating complementary feature sources and decision layers.

Advanced techniques discussed in Section 6 mark a decisive shift toward representation-centric intelligence. Deep learning–based representation learning, graph-based modeling, hardware-assisted detection, and adaptive learning paradigms significantly enhance generalization and evasion resistance. However, these gains are often accompanied by increased system complexity, computational overhead, and reduced transparency, underscoring the importance of explainable and interpretable detection mechanisms. Adversarial studies further reveal that accuracy alone is insufficient; detection systems must be evaluated under realistic threat models that include cooperative attacks, delayed execution, and deliberate feature manipulation.

Despite substantial progress, several open challenges remain and define promising directions for future research:

- Pre-Encryption and Ultra-Early Detection: While many approaches achieve high post-execution accuracy, preventing irreversible encryption remains the primary operational objective. Future work must prioritize detection within very short execution windows, leveraging contextual sequence modeling, hardware-level signals, and informed hybrid strategies to enable timely intervention.
- Concept Drift–Aware and Long-Term Deployment: Ransomware behaviors evolve continuously, rendering static models obsolete. Drift-aware feature engineering, adaptive classifiers, and streaming analytics must be integrated as first-class components rather than post hoc enhancements, with standardized evaluation protocols to assess long-term robustness.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 06 , June - 2026

TABLE VIII. COMPARISON OF REPRESENTATIVE RANSOMWARE-DETECTION SURVEYS ALONG THE DIMENSIONS EMPHASIZED IN THIS REVIEW.

Survey	Year	Platform scope	Pre-encryption & early detection	Concept drift / XAI / adversarial	Systematic (PRISMA + RQs)
Urooj et al. [1]	2022	Windows, Android, IoT, cloud	Partial	Limited	No
Or-Meir et al. [98]	2019	Generic malware (cross-OS)	No	Partial	No
Razaulla et al. [97]	2023	Cross-platform	Limited	Partial	No
This survey	2026	Windows, Android, Linux, IoT, cloud, cross-platform	Yes (dedicated)	Yes (dedicated)	Yes (PRISMA + RQs)

- Explainability and Operational Trust: As detection systems increasingly rely on deep and hybrid models, explainable artificial intelligence (XAI) is essential for trust, accountability, and forensic usability. Future research should focus on explainable-by-design architectures and decision fusion strategies that preserve transparency without sacrificing detection performance.
- Cross-Platform and Generalization-Driven Detection: Windows-centric solutions dominate the literature, while Linux, IoT, cloud-native, and cross-platform ransomware remain comparatively underexplored. Platform-agnostic representations, semantic embeddings, and knowledge-driven models offer promising paths toward generalized ransomware defense.
- Adversarial Robustness and Threat-Aware Evaluation: Adversarial manipulation and cooperative evasion attacks expose fundamental weaknesses in many existing detectors. Future systems must incorporate adversarial training, cross-process correlation, and threat-aware benchmarking to ensure resilience against adaptive attackers.
- Scalable and Lightweight Architectures: Deployment feasibility remains a critical barrier, particularly in enterprise, cloud, and resource-constrained environments. Research should emphasize lightweight hybridization, selective deep inference, and hardware–software co-design to balance detection strength with operational cost.
- Threat-Informed and ATT&CK-Aligned Detection: Most detectors are still assessed as binary classifiers rather than against the adversary behaviors they must counter. Mapping detected activity to MITRE ATT&CK tactics and techniques—and reporting coverage at that level—would enable threat-informed evaluation, clarify which attack stages each method actually addresses, and improve interoperability with operational defense tooling.

Limitations of this review. Several limitations should be acknowledged. First, the primary structured search was conducted in Scopus; although it indexes the major venues in this area and was complemented by backward and forward citation tracking, relevant work indexed elsewhere or released only as preprints may have been missed. Second, the review window (2017–2026) privileges recent contributions and may underrepresent earlier foundational studies beyond those reintroduced here. Third, the comparative assessments—including the cross-survey comparison in Table VIII and the performance figures discussed throughout—rest on the descriptions reported by the original studies, which differ in datasets, metrics, and evaluation rigor and were not independently re-executed. Finally, in such a fast-moving field, some very recent methods may have appeared after the search cutoff. We have sought to mitigate these constraints through explicit eligibility criteria, a PRISMA-based selection process, and a deliberately critical reading of reported results.

In summary, ransomware detection research has matured from isolated static and dynamic techniques into a rich ecosystem of adaptive, multi-layer, and intelligence-driven defenses. Continued progress will depend on unifying early detection, robustness, explainability, and scalability within coherent and deployable frameworks. By addressing these challenges, future ransomware detection systems can move beyond benchmark performance toward sustained, real-world protection in increasingly hostile and dynamic cyber environments.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 06 , June - 2026

REFERENCES

[1] U. Urooj, B. A. S. Al-Rimy, A. Zainal, F. A. Ghaleb, and M. A. Rassam, "Ransomware Detection Using the Dynamic Analysis and Machine Learning: A Survey and Research Directions," Applied Sciences (Switzerland), Article vol. 12, no. 1, 2022, Art no. 172, doi: 10.3390/app12010172.

[2] M. U. R. Shaikh et al., "Pre-Encryption Ransomware Detection (PERD) Taxonomy, and Research Directions: Systematic Literature Review," International Journal of Advanced Computer Science and Applications, Article vol. 15, no. 9, pp. 165–184, 2024, doi: 10.14569/IJACSA.2024.0150917.

[3] D. T. Nguyen and S. Lee, "LightGBM-based Ransomware Detection using API Call Sequences," International Journal of Advanced Computer Science and Applications, Article vol. 12, no. 10, pp. 138–146, 2021, doi: 10.14569/IJACSA.2021.0121016.

[4] A. N. Jahromi, S. Hashemi, A. Dehghantanha, R. M. Parizi, and K. K. R. Choo, "An Enhanced Stacked LSTM Method with No Random Initialization for Malware Threat Hunting in Safety and Time-Critical Systems," IEEE Transactions on Emerging Topics in Computational Intelligence, Article vol. 4, no. 5, pp. 630–640, 2020, Art no. 9122024, doi: 10.1109/TETCI.2019.2910243.

[5] M. Sukul, S. A. Lakshmanan, and R. Gowtham, "Automated Dynamic Detection of Ransomware using Augmented Bootstrapping," 2022, pp. 787–794, doi: 10.1109/ICOEI53556.2022.9777099.

[6] M. A. Ayub, A. Siraj, B. Filar, and M. Gupta, "RWArmor: a static-informed dynamic analysis approach for early detection of cryptographic windows ransomware," International Journal of Information Security, Article vol. 23, no. 1, pp. 533–556, 2024, doi: 10.1007/s10207-023-00758-z.

[7] B. Jethva, I. Traoré, A. Ghaleb, K. Ganame, and S. Saad, "Multilayer ransomware detection using grouped registry key operations, file entropy and file signature monitoring," Journal of Computer Security, Article vol. 28, no. 3, pp. 337–373, 2020, doi: 10.3233/JCS-191346.

[8] F. Ceschin, M. Botacin, H. M. Gomes, F. Pinagé, L. S. Oliveira, and A. Gregio, "Fast & Furious: On the modelling of malware detection as an evolving data stream[Formula presented]," Expert Systems with Applications, Article vol. 212, 2023, Art no. 118590, doi: 10.1016/j.eswa.2022.118590.

[9] D. W. Fernando and N. Komninos, "FeSAD ransomware detection framework with machine learning using adaption to concept drift," Computers and Security, Article vol. 137, 2024, Art no. 103629, doi: 10.1016/j.cose.2023.103629.

[10] F. De Gaspari, D. Hitaj, G. Pagnotta, L. De Carli, and L. V. Mancini, "Evading behavioral classifiers: a comprehensive analysis on evading ransomware detection techniques," Neural Computing and Applications, Article vol. 34, no. 14, pp. 12077–12096, 2022, doi: 10.1007/s00521-022-07096-6.

[11] F. De Gaspari, D. Hitaj, G. Pagnotta, L. De Carli, and L. V. Mancini, "The Naked Sun: Malicious Cooperation Between Benign-Looking Processes," in Lecture Notes in Computer Science, 2020, vol. 12147 LNCS, pp. 254–274, doi: 10.1007/978-3-030-57878-7_13.

[12] S. K. Shaukat and V. J. Ribeiro, "RansomWall: A layered defense system against cryptographic ransomware attacks using machine learning," 2018, vol. 2018-January, pp. 356–363, doi: 10.1109/COMSNETS.2018.8328219.

[13] M. M. Hasan and M. M. Rahman, "RansHunt: A support vector machines based ransomware analysis framework with integrated feature set," 2017, vol. 2018-January, pp. 1–7, doi: 10.1109/ICCITECHN.2017.8281835.

[14] [P. Mohan Anand, P. V. Putrevu, and S. K. Shukla, "A Comprehensive API Call Analysis for Detecting Windows-Based Ransomware," 2022, pp. 337–344, doi: 10.1109/CSR54599.2022.9850320.

[15] B. M. Khammas, "Ransomware Detection using Random Forest Technique," ICT Express, Article vol. 6, no. 4, pp. 325–331, 2020, doi: 10.1016/j.icte.2020.11.001.

[16] S. Kesharwani and M. Malik, "From Behavior to Pixels: A Vision Transformer Approach for Android Ransomware Detection," in Lecture Notes in Networks and Systems, 2026, vol. 1600 LNNS, pp. 132–141, doi: 10.1007/978-3-032-03072-6_11.

[17] S. Satpathy and . K. Swain, "Graph-contrast ransomware detection (GCRD) with advanced feature selection and deep learning," Discover Computing, Article vol. 28, no. 1, 2025, Art no. 124, doi: 10.1007/s10791-025-09651-w.

[18] [18] S. Maniath, A. Ashok, P. Poornachandran, V. G. Sujadevi, A. U. P. Prem Sankar, and S. Jan, "Deep learning LSTM based ransomware detection," 2018, pp. 442–446, doi: 10.1109/RDCAPE.2017.8358312.

[19] S. S. Chen, Y. X. Wu, Y. X. Ho, P. P. Cheng, and C. Y. Sun, "RANsomCheck: A CNN-Transformer Model for Malware Detection Based on API Call Sequences," in Lecture Notes in Computer Science, 2026, vol. 15707 LNAI, pp. 116–127, doi: 10.1007/978-981-96-8892-0_10.

[20] S. Gulmez, A. Kakisim, and I. Sŏgukpinar, "XRan: Explainable deep learning-based ransomware detection using dynamic analysis," Computers and Security, Article vol. 139, 2024, Art no. 103703, doi: 10.1016/j.cose.2024.103703.

[21] R. A. Mowri, M. Siddula, and K. Roy, "Is iterative feature selection technique efficient enough? A comparative performance analysis of RFECV feature selection technique in ransomware classification using SHAP," Discover Internet of Things, Article vol. 3, no. 1, 2023, Art no. 21, doi: 10.1007/s43926-023-00053-2.

[22] M. Adnan Alvi and Z. Jalil, "XRGuard: A Model-Agnostic Approach to Ransomware Detection Using Dynamic Analysis and Explainable AI," IEEE Access, Article vol. 13, pp. 53159–53170, 2025, doi: 10.1109/ACCESS.2025.3553562.

[23] P. Mohan Anand, P. V. Putrevu, and S. K. Shukla, "HiPeR - Early Detection of a Ransomware Attack using Hardware Performance Counters," Digital Threats: Research and Practice, Article vol. 4, no. 3, 2023, Art no. 43, doi: 10.1145/3608484.

[24] G. O. Ganfure, C. F. Wu, Y. H. Chang, and W. K. Shih, "DeepWare: Imaging Performance Counters With Deep Learning to Detect Ransomware," IEEE Transactions on Computers, Article vol. 72, no. 3, pp. 600–613, 2023, doi: 10.1109/TC.2022.3173149.

[25] M. Aljabri et al., "Ransomware detection based on machine learning using memory features," Egyptian Informatics Journal, Article vol. 25, 2024, Art no. 100445, doi: 10.1016/j.eij.2024.100445.

[26] M. Alonso, A. Girones, J. J. Costa, E. Morancho, S. Di Carlo, and R. Canal, "Automatic linux malware detection using binary inspection and runtime opcode tracing," Microprocessors and Microsystems, Article vol. 120, 2026, Art no. 105237, doi: 10.1016/j.micpro.2025.105237.

[27] H. Zhang, X. Xiao, F. Mercaldo, S. Ni, F. Martinelli, and A. K. Arun Kumar, "Classification of ransomware families with machine learning based on N-gram of opcodes," Future Generation Computer Systems, Article vol. 90, pp. 211–221, 2019, doi: 10.1016/j.future.2018.07.052.

[28] B. Zhang, W. Xiao, X. Xiao, A. K. Arun Kumar, W. Zhang, and J. Zhang, "Ransomware classification using patch-based CNN and self-attention network on embedded N-grams of opcodes," Future Generation Computer Systems, Article vol. 110, pp. 708–720, 2020, doi: 10.1016/j.future.2019.09.025.

[29] S. Johnson, R. Gowtham, and A. R. Nair, "Ensemble Model Ransomware Classification: A Static Analysis-based Approach," in Lecture Notes in Networks and Systems, 2022, vol. 336, pp. 153–167, doi: 10.1007/978-981-16-6723-7_12.

[30] B. Yamany, M. S. Said Elsayed, A. D. Jurcut, N. Abdelbaki, and M. A. Azer, "A New Scheme for Ransomware Classification and Clustering Using Static Features," Electronics (Switzerland), Article vol. 11, no. 20, 2022, Art no. 3307, doi: 10.3390/electronics11203307.

[31] B. Yamany, M. S. Said Elsayed, A. D. Jurcut, N. Abdelbaki, and M. A. Azer, "A Holistic Approach to Ransomware Classification: Leveraging Static and Dynamic Analysis with Visualization," Information (Switzerland), Article vol. 15, no. 1, 2024, Art no. 46, doi: 10.3390/info15010046.

[32] K. Alissa et al., "Dwarf Mongoose Optimization with Machine-Learning-Driven Ransomware Detection in Internet of Things Environment," Applied Sciences (Switzerland), Article vol. 12, no. 19, 2022, Art no. 9513, doi: 10.3390/app12199513.

[33] K. Zirari, H. K. Kamal Idrissi, A. El-Yahyaoui, H. Bensaid, and A. En-Nouaary, "Deciphering Ransomware: Strategic API Usage and Behavioral Patterns for Advanced Detection Techniques," International Journal of Interactive Mobile Technologies, Article vol. 19, no. 10, pp. 199–221, 2025, doi: 10.3991/ijim.v19i10.49245.

[34] A. Gharib and A. Ghorbani, "DNA-Droid: A real-time android ransomware detection framework," in Lecture Notes in Computer

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 06 , June - 2026

Science, 2017, vol. 10394 LNCS, pp. 184–198, doi: 10.1007/978-3-319-64701-2_14.

[35] T. G. Gregory Paul and T. Gireesh Kumar, "A framework for dynamic malware analysis based on behavior artifacts," in Advances in Intelligent Systems and Computing, 2017, vol. 515, pp. 551–559, doi: 10.1007/978-981-10-3153-3_55.

[36] F. Cara, M. Scalas, G. Giacinto, and D. Maiorca, "On the feasibility of adversarial sample creation using the android system API," Information (Switzerland), Article vol. 11, no. 9, 2020, Art no. 433, doi: 10.3390/INFO11090433.

[37] M. Almousa, S. Basavaraju, and M. Anwar, "API-Based Ransomware Detection Using Machine Learning-Based Threat Detection Models," 2021, doi: 10.1109/PST52912.2021.9647816.

[38] K. S. Kader, M. T. H. Tahsin, M. S. Hossain, and H. S. Narman, "Ransomware Detection Using Binary Classification," 2021, pp. 979–984, doi: 10.1109/DASC-PICom-CBDCom-CyberSciTech52372.2021.00163.

[39] Q. Chen, S. R. Islam, H. Haswell, and R. A. Bridges, "Automated Ransomware Behavior Analysis: Pattern Extraction and Early Detection," in Lecture Notes in Computer Science, 2019, vol. 11933 LNCS, pp. 199–214, doi: 10.1007/978-3-030-34637-9_15.

[40] S. Usharani, P. M. Manju Bala, and M. M. J. Martina Jose Mary, "Dynamic analysis on crypto-ransomware by using machine learning: Gandcrab ransomware," in Journal of Physics: Conference Series, 2021, vol. 1717, 1 ed., doi: 10.1088/1742-6596/1717/1/012024.

[41] Y. A. Ahmed, B. Koçer, and B. A. S. Al-Rimy, "Automated Analysis Approach for the Detection of High Survivable Ransomware," KSII Transactions on Internet and Information Systems, Article vol. 14, no. 5, pp. 2236–2257, 2020, doi: 10.3837/tiis.2020.05.021.

[42] J. A. Herrera Silva and M. Hernández-Alvarez, "Dynamic Feature Dataset for Ransomware Detection Using Machine Learning Algorithms," Sensors, Article vol. 23, no. 3, 2023, Art no. 1053, doi: 10.3390/s23031053.

[43] N. Youssef, N. Elbaraway, and A. Elmaghraby, "Transformer-Based API Call Sequence Modeling for Dynamic Malware Detection," in Conference Proceedings - IEEE SOUTHEASTCON, 2025, pp. 494–500, doi: 10.1109/SoutheastCon56624.2025.10971600.

[44] T. L. Lin et al., "Ransomware Detection by Distinguishing API Call Sequences through LSTM and BERT Models," Computer Journal, Article vol. 67, no. 2, pp. 632–641, 2024, doi: 10.1093/comjnl/bxad005.

[45] D. Nahmias, A. Cohen, N. Nissim, and Y. Elovici, "TrustSign: Trusted Malware Signature Generation in Private Clouds Using Deep Feature Transfer Learning," in Proceedings of the International Joint Conference on Neural Networks, 2019, vol. 2019-July, doi: 10.1109/IJCNN.2019.8851841.

[46] M. A. Alvi, Z. Jalil, M. Imran, and M. Ayub, "RGV2: A Framework for crypto-ransomware detection using FastText and neural networks," Knowledge and Information Systems, Article vol. 68, no. 1, 2026, Art no. 21, doi: 10.1007/s10115-025-02661-6.

[47] W. A. Iwan, V. Suryani, and F. A. Yulianto, "Realtime Ransomware Detection Based on Host System Utilizing Machine Learning and Windowing Threshold," 2025, pp. 873–879, doi: 10.1109/ICoDSA67155.2025.11157409.

[48] J. von der Assen et al., "HyperDtct: Hypervisor-based Ransomware Detection using System Calls," 2025, pp. 308–313, doi: 10.1109/CSR64739.2025.11130172.

[49] J. Tan and M. Abdulnabi, "Enhancing Cyber Defense Strategies: A Triple-Filter Ransomware Interpretable Model (TriFRIM) for Ransomware Detection," 2025, doi: 10.1109/ICMCTC62214.2025.11196535.

[50] P. S. Nguyen, T. N. Huy, T. A. Tuan, P. D. Trung, and H. V. Long, "Hybrid feature extraction and integrated deep learning for cloud-based malware detection," Computers and Security, Article vol. 150, 2025, Art no. 104233, doi: 10.1016/j.cose.2024.104233.

[51] M. Hirano and R. Kobayashi, "Machine Learning Based Ransomware Detection Using Storage Access Patterns Obtained from Live-forensic Hypervisor," 2019, doi: 10.1109/IOTSMS48152.2019.8939214.

[52] M. Hirano, R. Hodota, and R. Kobayashi, "RanSAP: An open dataset of ransomware storage access patterns for training machine learning models," Forensic Science International: Digital Investigation, Article vol. 40, 2022, Art no. 301314, doi: 10.1016/j.fsidi.2021.301314.

[53] M. A. Ayub and A. Siraj, "Understanding the Behavior of Ransomware: An I/O Request Packet (IRP) Driven Study on Ransomware Detection against Execution Time," 2023, pp. 62–71, doi: 10.1109/CIC58953.2023.00018.

[54] S. Aurangzeb, R. N. Rais, M. Aleem, M. A. Islam, and M. A. Iqbal, "On the classification of Microsoft-Windows ransomware using hardware profile," PeerJ Computer Science, Article vol. 7, pp. 1–24, 2021, doi: 10.7717/peerj-cs.361.

[55] C. Woralert, C. Liu, and Z. Blasingame, "Feasibility of Scalable Performance Counter Malware Detection Framework Using LightGBM," 2025, pp. 55–63, doi: 10.1145/3768725.3768735.

[56] Z. Pan and Z. Shu, "Hardware-Assisted Ransomware Detection Using Automated Machine Learning," in Proceedings -Design, Automation and Test in Europe, DATE, 2025, doi: 10.23919/DATE64628.2025.10993115.

[57] M. S. Rosli, R. S. Abdullah, W. Yassin, and M. A. Faizal, "Ransomware behavior attack construction via graph theory approach," International Journal of Advanced Computer Science and Applications, Article no. 2, pp. 487–496, 2020, doi: 10.14569/ijacsa.2020.0110262.

[58] A. Mahboubi et al., "Ransomware Encryption Detection: Adaptive File System Analysis Against Evasive Encryption Tactics," in Lecture Notes in Computer Science, 2025, vol. 15660 LNCS, pp. 399–414, doi: 10.1007/978-981-96-9101-2_21.

[59] A. K. Upadhyay, P. Dubey, S. Gandhi, and S. Jain, "Ransomware Detection And Data Recovery," 2024, doi: 10.1109/ICEECT61758.2024.10738908.

[60] [60] A. Arfeen, M. Khan, O. Zafar, and U. Ahsan, "Process based volatile memory forensics for ransomware detection," Concurrency and Computation: Practice and Experience, Article vol. 34, no. 4, 2022, Art no. e6672, doi: 10.1002/cpe.6672.

[61] A. Ali-Gombe, S. Sudhakaran, R. Vijayakanthan, and G. G. Richard, "cRGB_Mem: At the intersection of memory forensics and machine learning," Forensic Science International: Digital Investigation, Article vol. 45, 2023, Art no. 301564, doi: 10.1016/j.fsidi.2023.301564.

[62] K. Ganame, M. A. Allaire, G. Zagdene, and O. Boudar, "Network Behavioral Analysis for Zero-Day Malware Detection – A Case Study," in Lecture Notes in Computer Science, 2017, vol. 10618 LNCS, pp. 169–181, doi: 10.1007/978-3-319-69155-8_13.

[63] A. O. Almashhadani, M. Kaiiali, S. Sakir Sezer, and P. O'Kane, "A Multi-Classifier Network-Based Crypto Ransomware Detection System: A Case Study of Locky Ransomware," IEEE Access, Article vol. 7, pp. 47053–47067, 2019, Art no. 8674751, doi: 10.1109/ACCESS.2019.2907485.

[64] H. Zuhair, A. Selamat, and O. Krejcar, "A multi-tier streaming analytics model of 0-day ransomware detection using machine learning," Applied Sciences (Switzerland), Article vol. 10, no. 9, 2020, Art no. 3210, doi: 10.3390/app10093210.

[65] C. Ng, S. Rajasegarar, L. Pan, F. Jiang, and L. Y. Zhang, "VoterChoice: A ransomware detection honeypot with multiple voting framework," Concurrency and Computation: Practice and Experience, Article vol. 32, no. 14, 2020, Art no. e5726, doi: 10.1002/cpe.5726.

[66] X. Yang, S. Ruan, Y. Yue, and B. Sun, "PETNet: Plaintext-aware encrypted traffic detection network for identifying Cobalt Strike HTTPS traffics," Computer Networks, Article vol. 238, 2024, Art no. 110120, doi: 10.1016/j.comnet.2023.110120.

[67] P. Bhandary and C. Nicholas, "A Behavioral Analysis of Ransomware in Active Directory: A Case Study of BlackMatter, Conti, LockBit, and Midnight," 2025, doi: 10.1109/ISDFS65363.2025.11012104.

[68] M. R. Ali Khan and A. Almulhem, "Behavioral and Propagation-Based Analysis of APT Attacks for Effective Attack Attribution," 2025, doi: 10.1109/ISDFS65363.2025.11012042.

[69] H. M. H. M. Bandara, K. M. N. Ayeshani, M. M. P. M. Kumari, D. M. S. T. Wijerathna, K. Y. Abeywardena, and A. Wijesooriya, "Stealth Eye: Behavioral Analysis for Fileless Malware Detection," 2025, doi: 10.1109/ISDFS65363.2025.11012086.

[70] X. A. Arputha Rathina, M. Aadil, and D. Monesh, "Ransomware Detection in Android using Machine Learning," 2024, pp. 180–185, doi: 10.1109/OCIT65031.2024.00040.

[71] V. Pahuja, A. Khanna, and I. Sharma, "RansomSheild: Novel Framework for Effective Data Recovery in Ransomware Recovery Process," 2024, pp. 240–245, doi: 10.1109/ICBDML60909.2024.10577365.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 06 , June - 2026

[72] A. O. Almashhadani, D. Carlin, M. Kaiiali, and S. Sakir Sezer, "MFMCNS: a multi-feature and multi-classifier network-based system for ransomworm detection," Computers and Security, Article vol. 121, 2022, Art no. 102860, doi: 10.1016/j.cose.2022.102860.

[73] A. Mumtaz, V. Garg, and S. Sharma, "A Machine Learning Framework For Efficient Malware Detection and Classification," 2025, pp. 789–794, doi: 10.1109/CICTN64563.2025.10932393.

[74] E. B. Karbab, M. Debbabi, and A. Derhab, "SwiftR: Cross-platform ransomware fingerprinting using hierarchical neural networks on hybrid features," Expert Systems with Applications, Article vol. 225, 2023, Art no. 120017, doi: 10.1016/j.eswa.2023.120017.

[75] Y. A. Abid, J. Wu, M. Farhan, and T. Ahmad, "ECMT Framework for Internet of Things: An Integrative Approach Employing In-Memory Attribute Examination and Sophisticated Neural Network Architectures in Conjunction With Hybridized Machine Learning Methodologies," IEEE Internet of Things Journal, Article vol. 11, no. 4, pp. 5867–5886, 2024, doi: 10.1109/JIOT.2023.3312152.

[76] D. W. Fernando and N. Komninos, "FeSA: Feature selection architecture for ransomware detection under concept drift," Computers and Security, Article vol. 116, 2022, Art no. 102659, doi: 10.1016/j.cose.2022.102659.

[77] L. B. Bhagwat and B. M. Patil, "Behavioural analysis and results of malware and ransomware using optimal behavioural feature set," International Journal of Information and Computer Security, Article vol. 23, no. 1, pp. 57–78, 2024, doi: 10.1504/IJICS.2024.136719.

[78] M. Davidian, M. Kiperberg, and N. Vanetik, "Early Ransomware Detection with Deep Learning Models," Future Internet, Article vol. 16, no. 8, 2024, Art no. 291, doi: 10.3390/fi16080291.

[79] D. Sgandurra, L. Muñoz-González, R. Mohsen, and E. C. Lupu, “Automated Dynamic Analysis of Ransomware: Benefits, Limitations and use for Detection,” arXiv preprint, arXiv:1609.03020, 2016.

[80] A. Kharaz, S. Arshad, C. Mulliner, W. Robertson, and E. Kirda, “UNVEIL: A Large-Scale, Automated Approach to Detecting Ransomware,” in Proc. 25th USENIX Security Symposium, 2016, pp. 757–772.

[81] A. Continella et al., “ShieldFS: A Self-Healing, Ransomware-Aware Filesystem,” in Proc. 32nd Annual Conf. on Computer Security Applications (ACSAC), 2016, pp. 336–347, doi: 10.1145/2991079.2991110.

[82] N. Scaife, H. Carter, P. Traynor, and K. R. B. Butler, “CryptoLock (and Drop It): Stopping Ransomware Attacks on User Data,” in Proc. IEEE 36th Int. Conf. on Distributed Computing Systems (ICDCS), 2016, pp. 303–312, doi: 10.1109/ICDCS.2016.46.

[83] E. Kolodenker, W. Koch, G. Stringhini, and M. Egele, “PayBreak: Defense Against Cryptographic Ransomware,” in Proc. ACM Asia Conf. on Computer and Communications Security (ASIACCS), 2017, pp. 599–611, doi: 10.1145/3052973.3053035.

[84] S. Malik, B. Shanmugam, K. Krishnan, and S. Azam, "Critical Feature Selection for Machine Learning Approaches to Detect Ransomware," International Journal of Computing and Digital Systems, Article vol. 11, no. 1, pp. 1167–1176, 2022, doi: 10.12785/ijcds/110195.

[85] [85] M. Jaya and M. F. A. Ab Razak, "Dynamic Ransomware Detection for Windows Platform Using Machine Learning Classifiers," International Journal on Informatics Visualization, Article vol. 6, no. 2, pp. 469–474, 2022, doi: 10.30630/joiv.6.2-2.1093.

[86] M. M. Mahinderjit Singh, K. Selvaraj, and Z. Wei, "Enhanced detection of android ransomware families using machine learning and network traffic analysis," Bulletin of Electrical Engineering and Informatics, Article vol. 14, no. 4, pp. 2987–2996, 2025, doi: 10.11591/eei.v14i4.9485.

[87] Z. Abdullah, F. W. Muhadi, M. M. Mohd Saudi, I. R. A. Hamid, and C. F. M. Mohd Foozy, "Android Ransomware Detection Based on Dynamic Obtained Features," in Advances in Intelligent Systems and Computing, 2020, vol. 978 AISC, pp. 121–129, doi: 10.1007/978-3-030-36056-6_12.

[88] S. Sharmeen, Y. A. Ahmed, S. Huda, B. Koçer, and M. M. Hassan, "Avoiding Future Digital Extortion through Robust Protection against Ransomware Threats Using Deep Learning Based Adaptive Approaches," IEEE Access, Article vol. 8, pp. 24522–24534, 2020, Art no. 8976152, doi: 10.1109/ACCESS.2020.2970466.

[89] Y. T. Bau, Y. H. Choo, and C. L. Goh, "Android Malware Multiclass Classification using Machine Learning: Evaluating the Performance of Random Forest, Artificial Neural Network, and Convolutional Neural Network," Journal of Logistics, Informatics and Service Science, Article vol. 11, no. 10, pp. 1–19, 2024, doi: 10.33168/JLISS.2024.1001.

[90] M. Alonso, J. J. Costa, and E. Morancho, "Malware Detection on Linux Using Runtime Opcode Tracing," in Proceedings - IEEE International Symposium on Defect and Fault Tolerance in VLSI and Nanotechnology Systems, DFT, 2024, doi: 10.1109/DFT63277.2024.10753555.

[91] M. Maghanaki, S. Keramati, F. F. Chen, and M. Shahin, "Generation of a Multi-Class IoT Malware Dataset for Cybersecurity," Electronics (Switzerland), Article vol. 14, no. 21, 2025, Art no. 4196, doi: 10.3390/electronics14214196.

[92] G. Saranya, K. Kumaran, A. Misra, and M. Mishra, "Context-Aware Malware Scoring: A Multi-Model Ensemble Approach using Structured and Unstructured Data," 2025, pp. 53–59, doi: 10.1109/ICICV64824.2025.11085912.

[93] R. Jain and N. Nihalani, "Botnet Detection in Distributed Network Using Machine Learning- A Detailed Review," 2024, pp. 888–895, doi: 10.1109/IC3SE62002.2024.10593476.

[94] P. Wang, H. C. Li, H. C. Lin, W. H. Lin, and N. Z. Xie, "A Transductive Zero-Shot Learning Framework for Ransomware Detection Using Malware Knowledge Graphs," Information (Switzerland), Article vol. 16, no. 6, 2025, Art no. 458, doi: 10.3390/info16060458.

[95] S. Danya, G. Gokulraj, N. Maheswaran, M. Pradeep Kumar, and S. Bose, "Blockchain-Enabled Ransomware Detection: A Hybrid Model Combining Behavioral Analysis and Machine Learning," 2025, doi: 10.1109/RMKMATE64874.2025.11042769.

[96] Y. A. Ahmed, B. Koçer, S. Huda, B. A. S. Al-Rimy, and M. M. Hassan, "A system call refinement-based enhanced Minimum Redundancy Maximum Relevance method for ransomware early detection," Journal of Network and Computer Applications, Article vol. 167, 2020, Art no. 102753, doi: 10.1016/j.jnca.2020.102753.

[97] S. Razaulla, C. Fachkha, C. Markarian, A. Gawanmeh, W. Mansoor, B. C. M. Fung, and C. Assi, “The Age of Ransomware: A Survey on the Evolution, Taxonomy, and Research Directions,” IEEE Access, vol. 11, pp. 40698–40723, 2023, doi: 10.1109/ACCESS.2023.3268535.

[98] O. Or-Meir, N. Nissim, Y. Elovici, and L. Rokach, “Dynamic Malware Analysis in the Modern Era—A State of the Art Survey,” ACM Computing Surveys, vol. 52, no. 5, 2019, Art no. 88, doi: 10.1145/3329786.

[99] K. Zirari, H. Kamal Idrissi, H. Bensaid, and A. El-Yahyaoui, “Novel Dataset and AI Benchmarks for Behavior-based Ransomware Detection,” International Journal of Intelligent Engineering and Systems, vol. 18, no. 11, 2025, doi: 10.22266/ijies2025.1231.19.