# VERA: A Dual-Validated, Behaviorally-Verified Repository of 15,000+ Raw Windows Ransomware Binaries

Garvit Agarwal*, Nithish Vasanth*, Seunghyun Cho*, Yousef Mohammed Y. Alomayri*,
Noah D. Pumphrey*, Feng Li*, Yucheng Xie†, Xukai Zou‡
*Purdue University, West Lafayette, Indiana, USA
{agarw347, nvasant, cho704, yalomayr, npumphr, fengli}@purdue.edu

†Yeshiva University, New York, USA
yucheng.xie@yu.edu
‡Indiana University, Indianapolis, Indiana, USA
xzou@iu.edu

***Abstract*—Reliable ransomware detection research is hindered by a scarcity of high-quality, reproducible data. Public repositories often suffer from label noise where generic malware, benign files, or corrupted binaries are misclassified. We introduce VERA (Validated Execution-based Ransomware Analysis), a repository of behaviorally verified raw Windows ransomware constructed using a comprehensive Dual-Validation Methodology. We processed 40,552 candidates with VirusTotal consensus and subjected them to dynamic verification in the CAPE Sandbox. This process revealed that 22,837 samples were dynamically silent due to defunct infrastructure or incompatibility with modern Windows environments. We segregated these into a secondary VERA-Latent dataset to preserve the comprehensive pool while cleaning the primary set.**

**For the remaining 15,448 samples in the primary VERA-Active dataset, we ensured family label purity by cross-referencing static AVClass, a malware labeling tool, consensus with unsupervised behavioral clustering of dynamic execution traces. This process explicitly prunes samples with conflicting static and dynamic identities. Benchmarking demonstrates the high quality of VERA-Active. A standard XGBoost classifier achieves $>$ 99.9% specificity, distinguishing VERA samples from the BODMAS general malware dataset, confirming the isolation of high-fidelity ransomware signatures distinct from general malicious noise. By releasing raw binaries and execution logs, VERA offers a standardized benchmark for reproducible research.**

***Index Terms*—Ransomware, Dataset, Machine Learning Security, Dynamic Analysis, Reproducibility**

## I. Introduction

Ransomware is a type of cyberthreat that encrypts the victim's data, severely compromising financial stability as well as the confidentiality and security of connected devices. It has evolved into a systemic threat affecting entire infrastructures. In the 2023 IC3 report, 2,825 cases and more than $59.6 million in damages from ransomware were reported [1]. In response, both defense and attack mechanisms have escalated. In particular, advances in machine learning (ML) have expanded their application in ransomware detection and classification.

The VERA dataset and associated analysis code are publicly available at: https://doi.org/10.5281/zenodo.17904705

ML has indeed contributed significantly to this field. However, these contributions rely heavily on the quality and label of training data. Several limitations exist in currently published ransomware datasets. Many do not contain up-to-date attack mechanisms, the number of ransomware families is often bounded, and raw binaries are frequently omitted due to liability concerns, leaving researchers with only pre-extracted features. Most importantly, there is often a lack of rigorous verification of ground-truth labels. In particular, most existing datasets do not presuppose verification of actually viable ransomware and do not structurally rule out the possibility of large-scale inclusion of non-active samples from a dynamic execution perspective. Accordingly, a trained model might work well in theory but fail in real-world scenarios where labels are noisy or samples are misclassified.

A critical but underexplored issue in ransomware research is *behavioral viability*. A large fraction of publicly labeled ransomware samples either fail to execute or remain behaviorally silent in realistic environments due to defunct command-and-control infrastructure, delayed triggers, or environment-dependent execution checks. Consequently, relying solely on static labels from external antivirus engines may introduce a substantial number of dormant or non-viable samples into ransomware corpora. Existing datasets typically conflate such samples with actively encrypting ransomware, introducing significant noise into behavior-based learning tasks.

Through this paper, we address these gaps by making four main contributions:

1) **Behavioral Viability as a First-Class Dataset Property:** We demonstrate that a majority of publicly labeled ransomware samples (**58.2%**) are behaviorally silent under dynamic execution. We explicitly separate these samples into a distinct **VERA-Latent** partition, enabling controlled experimentation on viable versus non-viable ransomware.
2) **Dual-Validation Framework for Ransomware Ground Truth and Behavioral Viability:** We

introduce a two-stage validation pipeline that combines static multi-vendor consensus (VirusTotal + AVClass) with dynamic behavioral verification in the CAPE Sandbox. Samples exhibiting conflicting static and dynamic identities are explicitly pruned or isolated.

3) **Family-Level Label Integrity via Cross-View Consistency:** We establish reliable ransomware family labels by first constructing a high-confidence consensus set using agreement across multiple static labeling tools (AVClass, Euphony, and ClarAVy), and then propagating labels to unresolved samples using behavioral similarity learned from dynamic execution traces.
4) **A Reproducible, Artifact-Rich Dataset Release:** We release raw binaries, sandbox execution logs, extracted features, and validation scripts to enable reproducible research and systematic evaluation.

Our experiments confirmed that using high-quality data with dual-verification of the active status is essential for increasing the reliability of the classification model, rather than simply using a large amount of data quantitatively. By providing only valid behavioral data that occurs in real attack environments, we expect **VERA** to overcome the limitations of existing datasets and provide a more realistic research environment.

The rest of this paper is organized as follows. Section II reviews related work on ransomware datasets and ground-truth labeling. Section III describes the dataset construction pipeline, including dynamic analysis, behavioral status definitions, and the dual-validation framework. Section IV introduces the VERA dataset and summarizes its composition and released artifacts. Section V details the family-level label unification methodology based on static label triangulation and behavioral propagation. Section VI evaluates the distinctiveness of VERA through ransomware versus general malware classification experiments. Section VII discusses future research directions, and Section VIII concludes the paper.

## II. Related Work

### A. Ransomware Datasets

Prior work on ML-based malware detection has repeatedly highlighted the lack of large-scale, openly available benchmark datasets suitable for reproducible and fair evaluation. Legal concerns, the risk of malware redistribution, and institutional security policies significantly limit the public release of raw malicious binaries, especially for ransomware [2]. As a result, many existing datasets rely on pre-extracted features or static labels without guarantees of behavioral correctness under execution.

*Static malware datasets.* Several large-scale datasets support static malware detection research. *EMBER* provides a widely used benchmark for malicious Windows PE detection through pre-extracted static features and labels, while withholding raw binaries [3]. Although *EMBER 2024* expands coverage and accounts for concept drift [4], it remains constrained to predefined feature representations, limiting flexibility for feature learning and deep-learning approaches that operate directly on raw binaries. Importantly, EMBER does not distinguish ransomware from other malware classes, nor does it validate whether samples exhibit active malicious behavior at execution time.

Other large-scale datasets such as *BODMAS* [5] and *SOREL-20M* [6] provide millions of raw PE files for general malware research. While these datasets enable studies at unprecedented scale, they do not perform behavioral verification, ransomware-specific curation, or family-level ground truth validation. Consequently, samples labeled as ransomware may include dormant, misclassified, or non-viable binaries.

### B. Behavioral Taxonomies and Dataset Limitations

To better characterize ransomware behavior beyond static indicators, several studies have proposed behavioral taxonomies. The *Marauder Map* framework decomposes ransomware lifecycles into staging, encryption, and extortion phases [2]. Such taxonomies provide valuable analytical structure but do not offer executable datasets or behaviorally verified ground truth. As a result, most existing datasets conflate actively encrypting ransomware with samples that are dormant, environment-gated, or execution-inert.

### C. Ground Truth Verification and Family Labeling

Establishing reliable ground truth for ransomware families remains a persistent challenge. Large-scale malware repositories and prior studies commonly rely on antivirus detections aggregated through platforms such as *VirusTotal*, which provide multi-vendor verdicts but suffer from inconsistent and noisy naming conventions (for example, one vendor labeling a sample “WannaCry” while another uses “WanaCrypt0r”). To mitigate this issue, tools such as *AVClass* [7] and its successor *AVClass2* [8] aggregate VirusTotal vendor labels using alias resolution and voting schemes to infer a consensus family assignment.

### D. Family Label Unification Tools

Several approaches aim to normalize antivirus labels into canonical family names. *AVClass* and *AVClass2* apply alias rules and majority voting to infer a single family assignment. *Euphony* [9] models label co-occurrence structure using a dependency graph to reduce the influence of outlier vendor tokens. More recently, *ClarAVy* [10] applies NLP-based normalization to separate specific family identifiers from generic malware descriptors.

While these approaches reduce naming inconsistency, they remain fundamentally *static* and can disagree in the presence of sparse detections, generic labels, or mixed-family detections. None of these tools account for whether a sample exhibits ransomware behavior during execution, motivating approaches that combine label unification with behavioral evidence.

### E. Dynamic Behavioral Datasets

To address the limitations of static analysis, several datasets incorporate dynamic behavioral signals. *ElderAN* executes

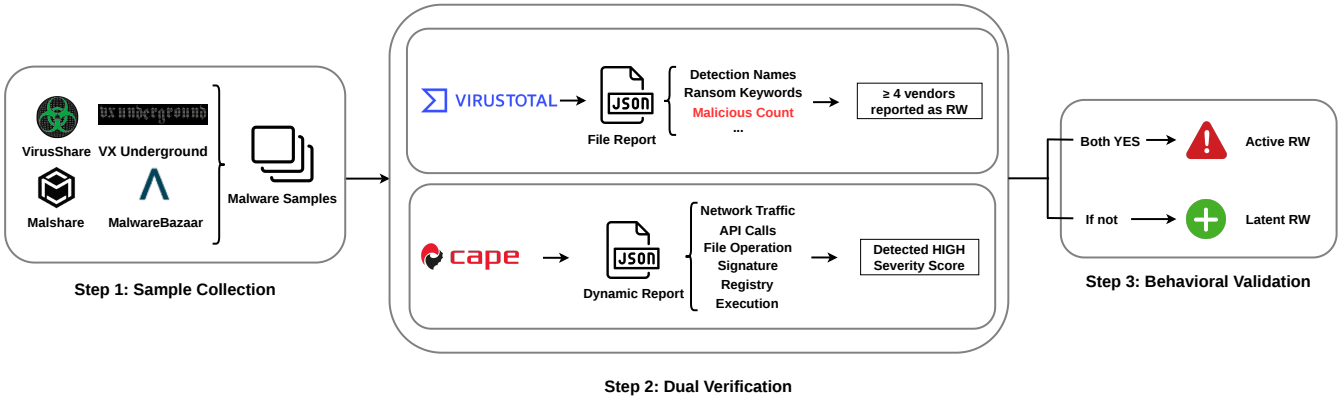


Fig. 1. Dataset Construction Overview

ransomware samples in Cuckoo Sandbox to extract runtime indicators for early crypto-ransomware detection [11], but is limited in size and family diversity. *RanSAP* captures low-level storage access patterns using a thin hypervisor [12], and its successor *RanSMAP* extends this to memory access patterns across multiple hardware configurations [13].

Similarly, *MLRan* proposes a behavior-based ransomware dataset spanning 64 families with improved balance [14], while other recent releases offer mixed benign and malicious behavioral samples [15]. Despite these advances, existing dynamic datasets often remain limited in scale, do not release raw binaries alongside execution artifacts, or lack multi-vendor family label integration.

### F. Ransomware Detection Methods

Parallel to dataset development, ransomware detection research increasingly leverages behavioral and sequence-based features. Prior work analyzes system call sequences to identify contextual execution patterns indicative of ransomware [16], while survey studies highlight the complementary roles of static, dynamic, and network features in detection pipelines [17].

More recent methods explore advanced architectures, including transformer-based models for family classification from dynamic API call traces [18], image-based representations of execution behavior [19], and ensemble frameworks such as RanDEL [20]. These approaches critically depend on high-quality behavioral datasets with reliable ground truth.

Existing ransomware datasets and detection studies exhibit a trade-off between scale, behavioral fidelity, and label reliability. Large-scale corpora primarily rely on static features or antivirus-derived labels, while dynamic ransomware datasets often remain limited in size, family diversity, or artifact availability. In addition, family label unification approaches are largely static and do not explicitly verify whether samples exhibit ransomware behavior at execution time. As a result, many datasets conflate actively encrypting ransomware with dormant or execution-inert samples, introducing noise into behavior-based analysis and learning tasks. VERA addresses this gap by integrating dynamic execution validation with multi-tool family label unification, while releasing raw binaries and execution artifacts to support reproducible, behavior-aware ransomware research.

## III. Dataset Construction Pipeline

We constructed the VERA repository using a multi-stage pipeline designed to filter low-quality noise and ensure behavioral fidelity. As illustrated in Fig. 1, our methodology proceeds from (i) sample collection, to (ii) dual verification via static consensus and dynamic execution in a controlled environment, followed by (iii) automated behavioral validation. Samples that satisfy both static and dynamic verification criteria are designated as *Active Ransomware*. These samples subsequently undergo family-level classification as described in Section V. Samples that fail either verification stage are retained as *Latent Ransomware* and excluded from family analysis.

### A. Data Collection

We collected an initial set of 44,678 candidate samples from widely used malware repositories, including VX Underground [21], VirusShare [22], MalwareBazaar [23], and Malshare [24], spanning the period from 2019 to 2024. These repositories label samples as malicious, and often as ransomware, based on community submissions, antivirus detections, or threat intelligence feeds. However, such labels are not accompanied by guarantees of behavioral correctness or execution viability, and they frequently lack evidence that the samples actively exhibit ransomware behavior under realistic execution environments. As a result, this initial corpus represents a set of *ransomware-labeled candidates* rather than a verified ransomware ground truth.

### B. Static Analysis Environment

As the first verification stage in our dual-validation pipeline, we perform static consensus filtering using multi-vendor antivirus reports obtained from VirusTotal [25] using our Academic API. For each collected sample, we retrieve the corresponding VirusTotal file report, which aggregates detection

results, malware family names, and keyword indicators from a large set of commercial antivirus engines.

A sample is considered to satisfy the static verification criterion if it is independently flagged as ransomware by at least four distinct antivirus vendors. This threshold is chosen to reduce the impact of noisy or overly generic detections from individual engines while retaining samples with broad agreement across vendors. Detection names and keyword matches associated with ransomware are used solely to establish consensus, rather than to infer behavioral correctness.

Samples that fail to achieve sufficient multi-vendor agreement are not discarded; instead, they are retained for dynamic analysis, where execution behavior ultimately determines their classification as *Active* or *Latent* ransomware.

### C. Dynamic Analysis Environment

All collected samples were executed in a custom dynamic analysis environment based on the CAPE Sandbox [26]. To capture the full lifecycle of the ransomware, each sample was executed for a duration of 3 minutes. The sandbox was configured to simulate a realistic user environment, preventing common evasion techniques that check for virtualization artifacts. This environment served as the generator for the raw execution logs ($L$) required for our validation framework.

### D. Validation Framework

To guarantee the reliability of our dataset, we implemented a rigorous automated validation pipeline that operates exclusively on dynamic behavioral artifacts captured during execution. Unlike static analysis, which can be evaded through obfuscation or packing, our approach assesses whether a sample exhibits concrete ransomware behavior when executed in a realistic environment.

We adapt the behavioral taxonomy proposed by Hou et al. [2] to engineer a deterministic classification algorithm. The resulting validation framework models the canonical ransomware lifecycle observed across modern families, progressing from environment assessment to data destruction and extortion.

*1) Indicator Extraction:* Our validation logic aggregates runtime events into three distinct operational stages. A sample must exhibit high-confidence indicators from these stages to be classified as valid ransomware.

- **Stage I: System Profiling and Staging.** We monitor for "pre-attack" behaviors where the binary assesses the host environment. This includes querying specific cryptographic APIs (e.g., `Device\CNG`) and enumerating recovery tools (e.g., `WinREAgent`) [2]. Additionally, we flag network discovery commands (e.g., `systeminfo`, `netstat`) used to map the local infrastructure.
- **Stage II: Recovery Inhibition and Encryption.** This stage contains the most definitive indicators of malicious intent. We scan for attempts to sabotage restoration mechanisms, specifically the deletion of Volume Shadow Copies via `vssadmin` or `wmic`, a tactic observed in nearly 90% of active families [2]. Furthermore, we detect high-entropy write operations consistent with file encryption and the clearing of Windows Event Logs (`wevtutil cl`) to mask forensic footprints.
- **Stage III: Extortion and Propagation.** To verify modern double-extortion capabilities, we identify the presence of known data egress tools (e.g., `WinSCP`, `Rclone`) and unauthorized connections to SMB (Port 445) or WSDAPI (Port 5357) [2]. The generation of a ransom note—files containing keywords such as *decrypt* or *instruction*—serves as a terminal state indicator for successful execution.

*2) Automated Classification Logic:* The classification is performed by the algorithm detailed in Algorithm 1. The procedure parses the raw sandbox logs ($L$) and extracts a set of weighted behavioral signatures ($S$).

**Algorithm 1** Ransomware Classification Pipeline

**Require:** $L$: Raw Execution Logs from Sandbox
**Ensure:** $Class$: Final Category (*Active*, *Latent*, *Failed*)

$Status \leftarrow$ VALIDATEEXECUTION($L$)
**if** $Status$ is **Error or** $L$ is **Empty then**
  **return** *Failed*
**end if**
$Signatures \leftarrow \emptyset$
$Signatures \leftarrow Signatures \cup$ SCANENVIRONMENT($L$) ▷ Stage I
$Signatures \leftarrow Signatures \cup$ DETECTDESTRUCTIVEOPS($L$) ▷ Stage II
$Signatures \leftarrow Signatures \cup$ CHECKDATAEGRESS($L$) ▷ Stage III
$HasNote \leftarrow$ DETECTRANSOMNOTE($L$)
**if** $HasNote$ is **True then**
  $Signatures \leftarrow Signatures \cup$ {"Ransom Note Generated"}
**end if**
**if** $\exists s \in Signatures$ such that $s$ is *HighConfidence* **then**
  **return** *Active Ransomware*
**else**
  **return** *Latent Ransomware*
**end if**

A behavioral signature is considered *HighConfidence* if it corresponds to irreversible or strongly indicative ransomware actions, such as backup deletion, multi-file high-entropy writes, or ransom note generation.

*3) Behavioral Status Definitions:* As specified in Algorithm 1, we categorize samples into three mutually exclusive behavioral states:

- **Active Ransomware:** Samples that produce at least one high-confidence ransomware lifecycle artifact during execution, such as volume shadow copy deletion, bursty high-entropy file writes across multiple user directories, or ransom note creation.
- **Latent Ransomware:** Samples that execute successfully but do not produce high-confidence ransomware artifacts

within the fixed execution window (3 minutes). These may be gated by unavailable infrastructure, delayed triggers, or environment-specific conditions.
- **Failed Execution:** Samples that crash, terminate immediately, or produce empty sandbox logs.

Samples categorized as *Failed Execution* are excluded from further analysis in this work. Importantly, latent samples are *not* considered benign binaries; rather, they are ransomware-labeled binaries whose malicious behavior could not be dynamically confirmed under our analysis conditions.

## IV. VERA Dataset

The result of the methodology described in Section III is **VERA**, a curated repository of ransomware binaries and their corresponding dynamic execution logs, constructed through rigorous behavioral validation.

### A. Dataset Composition

Applying our validation pipeline to the initial corpus revealed significant behavioral variance among samples. As shown in Table. I, while the majority of samples were valid executables, a substantial fraction did not exhibit observable ransomware behavior within the sandbox execution window. A smaller subset of samples failed to execute entirely due to crashes, early termination, or empty analysis logs and were therefore excluded from the released dataset.

All samples considered in this analysis satisfy the static multi-vendor consensus constraint described in Section III-B. Building on this requirement, as stated in Section III-D, classification as *Active Ransomware* additionally requires the sample to exhibit ransomware-specific malicious behavior. These Samples that meet the static consensus condition but fail to exhibit any of the ransomware-specific behavioral signals within the execution window are categorized as *Latent Ransomware*.

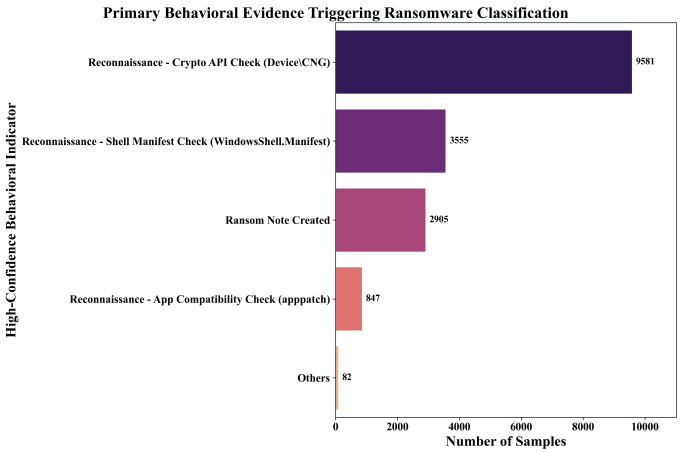


Fig. 2. Primary high-confidence behavioral indicators that triggered ransomware classification for Active samples. Each sample contributes its highest-severity indicator as defined by the validation logic in Section III.

Fig. 2 illustrates the dominant behavioral signals that triggered ransomware classification among **VERA-Active** samples. Notably, a large fraction of samples are classified based on early-stage indicators, such as cryptographic environment initialization and system reconnaissance, rather than terminal artifacts alone (i.e. ransom note creation). This demonstrates that the validation framework captures ransomware behavior across multiple phases of the attack lifecycle, including pre-encryption preparation, recovery inhibition, and extortion-related actions.

The distribution of highest-severity indicators further indicates that no single behavioral artifact dominates the Active set. While destructive actions such as volume shadow copy deletion and high-entropy file writes provide strong confirmation of malicious intent, a substantial portion of samples are validated before these terminal stages. This reflects real-world ransomware behavior, where execution paths may be interrupted by environment checks, delayed triggers, or partial execution, yet still expose identifiable malicious intent.

Furthermore, rather than discarding behaviorally silent samples, we retain and release all successfully executed binaries to support diverse research tasks, including static analysis, dormant ransomware detection, and delayed-trigger analysis. In contrast to **VERA-Active** samples, which exhibit at least one of the high-confidence behavioral indicators shown in Fig. 2, samples categorized as **VERA-Latent** fail to manifest any such indicators within the execution window, despite being structurally valid and statically labeled as ransomware. Samples that failed execution entirely (for example, crashes or empty logs) are excluded from the released dataset.

For all released samples, we provide a metadata CSV file (`vera_labels.csv`) that explicitly records the behavioral status assigned during validation:

1) **VERA-Active (15,448 samples):** Samples that satisfy the static multi-vendor consensus requirement described in Section III-B and exhibit at least one high-confidence ransomware behavioral indicator during dynamic execution, such as volume shadow copy deletion, high-entropy file writes, or ransom note generation.
2) **VERA-Latent (22,837 samples):** Samples that satisfy the same static consensus requirement but execute without producing any of the high-confidence behavioral indicators defined by the validation framework within the execution window, due to factors such as defunct infrastructure, delayed triggers, or environment-specific execution constraints.

The final composition of the dataset is summarized in Table I.

TABLE I
**VERA Dataset Composition**

| Category | Count |
|---|---|
| Total Collected Samples | 44,678 |
| **VERA-Active** | **15,448** |
| VERA-Latent | 22,837 |
| Excluded (Failed Execution) | 6,293 |

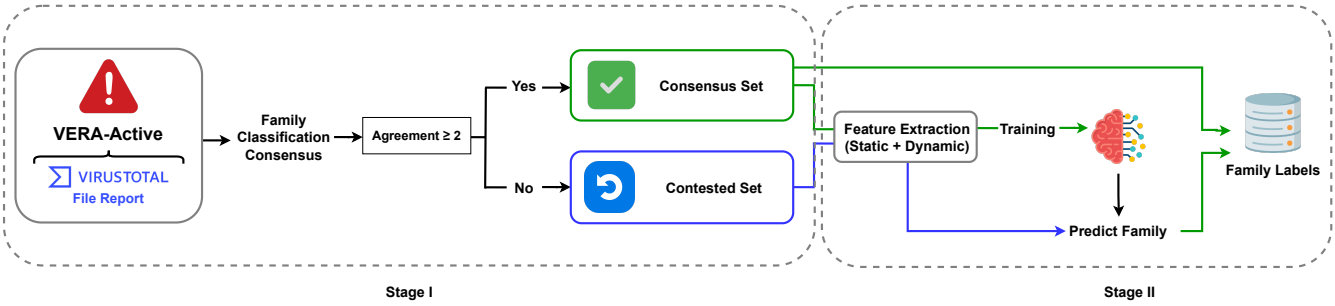


Fig. 3. Family Classification

### B. Feature Engineering

For every sample in the repository, we release the raw binary and the CAPE Sandbox JSON report. We also provide a pre-processed feature set extracted from the logs, which includes:

- **API Call Traces:** Sequences and frequencies of Windows API calls (such as `CreateFile`, `WriteFile`, `RegSetValue`).
- **File System Operations:** Summary of files created, deleted, read, and written, with a focus on file extension changes and entropy.
- **Registry Modifications:** Keys and values created, modified, or deleted, particularly those related to persistence.
- **Network Activity:** DNS requests, C2 communication patterns, and data exfiltration attempts.
- **Process Behavior:** Creation of new processes, thread injection, and other inter-process communication.

### C. Dataset Access and Reproducibility

To support reproducible research, **VERA** is released with standardized artifacts and metadata. Each sample is identified by its cryptographic hash and accompanied by its raw binary, full CAPE execution report, and a structured metadata record describing behavioral status and validation outcomes.

All feature extraction steps are deterministic and derived exclusively from released execution artifacts, enabling independent reproduction and extension of our results. No proprietary tools or manual annotations are required to regenerate the provided features or behavioral labels.

## V. Family Classification

After the collection of verified ransomware samples, we implemented a *Dual-Stage Labeling Pipeline* to ensure the integrity of the family labels. Fig 3 presents an overview of the family classification workflow.

### A. Stage I: Family Classification Consensus

We aggregated family attributions using three distinct malware labeling tools:

- **AVClass** [7]: Applies majority voting and vendor-specific alias rules.
- **Euphony** [9]: Constructs a label dependency graph to infer family names based on dominant centrality.
- **ClarAVy** [10]: Utilizes NLP to distinguish specific family nomenclature from generic malware tokens.

VirusTotal reports, as described in Section III-B, serve as the basis for extracting candidate family names. Each sample is labeled using the three family classification tools described above. When at least two of the three tools independently agree on the same family name, the sample is assigned that label and included in the **Consensus Set**.

Samples for which the tools fail to reach such agreement—including cases where all three tools assign different families, only one tool produces a family label, or none of the tools produce a valid family label—are grouped into the **Contested Set**.

TABLE II
Consensus Set vs. Contested Set After Stage I

| Status | Count | Percentage |
|---|---|---|
| Consensus Set | 12864 | 83.27% |
| Contested Set | 2584 | 16.73% |
| Total | 15448 | 100.00% |

### B. Stage II: Model Training & Predict Label

To accurately label the Contested Set, we treated the Consensus Set from Stage I as a trusted reference and used it as a training set for a Machine Learning model. This machine learning model captures characteristic static structure and execution behavior associated with known ransomware families, and applies this learned representation to infer family labels for samples in the Contested Set. To avoid unstable classes, the training set was constructed by selecting samples from families that had at least 50 samples, which resulted in 10,149 samples covering 41 ransomware families.

We extracted behavioral features from CAPE reports. Additionally, 2,912 static features from raw executable bytes were derived. To reduce noise and improve generalization, we dropped features with more than 40% missing values and removed near-zero-variance features where at least 98% of samples shared the same value, resulting in 2,893 numeric features after filtering.

To select an appropriate model for label propagation, the Consensus Set was partitioned into an 80% training split

TABLE III
MODEL SELECTION RESULTS

| Model | Accuracy |
|---|---|
| RandomForest | 0.9463 |
| ExtraTrees | 0.9458 |
| **XGBoost** | **0.9493** |
| CatBoost | 0.9458 |

(8,118 samples) and a 20% test split (2,030 samples). Four candidate classifiers were trained and evaluated on this split using identical feature representations. As summarized in Table III, XGBoost achieved the highest test accuracy ($\approx 0.95$) among the evaluated models. Based on this result, XGBoost was selected as the final model for propagating family labels to the Contested Set.

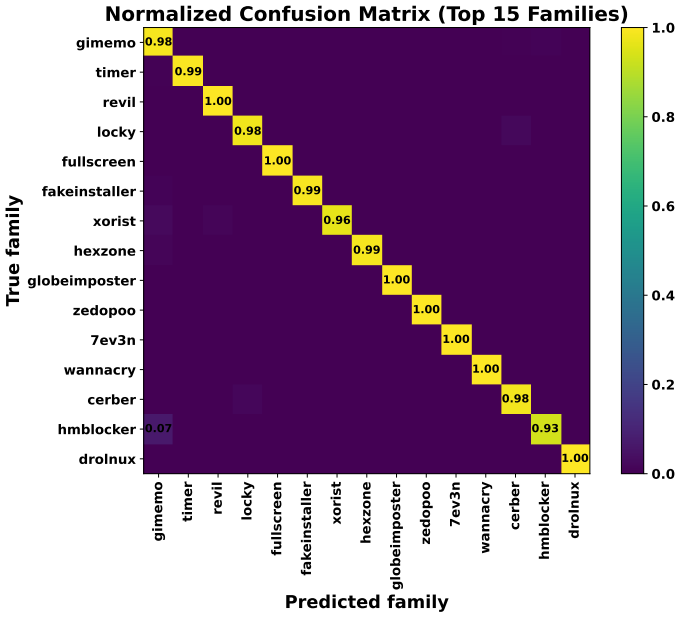


Fig. 4. Normalized confusion matrix for the top 15 families in the Consensus Set test split.

Figure 4 presents the normalized confusion matrix for the 15 most frequent families in the Consensus Set test split. The concentration of mass along the diagonal indicates that the selected features and model provide consistent separation among the major families, while the relatively small off-diagonal entries suggest that misclassifications are limited to a small number of behaviorally similar family pairs.

The same feature extraction procedure used for the Consensus Set was applied to samples in the Contested Set, and the resulting feature vectors were provided as input to the trained XGBoost model. The model assigns exactly one family label to each contested sample, yielding a complete family labeling for the dataset. The output of this stage is a CSV file containing the SHA256 hash and predicted family name for each previously contested sample. These predictions are subsequently merged back into the dataset, replacing unresolved entries and producing a single, unified family label per sample for downstream analysis.

After using the XGBoost classifier to infer labels for the Contested Set, we obtained a single unified family label for every sample in the corpus. The finalized labeling contains 15,448 samples spanning 671 unique ransomware families.

TABLE IV
TOP 20 RANSOMWARE FAMILIES BY SAMPLE COUNT

| Family | Count | Percent (%) |
|---|---|---|
| gimemo | 1605 | 10.39 |
| timer | 808 | 5.23 |
| revil | 650 | 4.21 |
| fullscreen | 629 | 4.07 |
| locky | 610 | 3.95 |
| cerber | 585 | 3.79 |
| zedopoo | 571 | 3.70 |
| pornostser | 528 | 3.42 |
| fakeinstaller | 498 | 3.22 |
| xorist | 453 | 2.93 |
| wannacry | 416 | 2.69 |
| hexzone | 414 | 2.68 |
| globeimposter | 406 | 2.63 |
| 7ev3n | 342 | 2.21 |
| shade | 339 | 2.19 |
| gandcrab | 334 | 2.16 |
| hmlocker | 309 | 2.00 |
| drolnux | 276 | 1.79 |
| birdele | 265 | 1.72 |
| zard | 253 | 1.64 |
| Top 20 Families | 10,291 | 66.62 |
| **Total (671 Families)** | **15,448** | **100** |

Table IV summarizes the distribution of the most prevalent ransomware families in the finalized labeling. The dataset is clearly quite diverse: *gimemo* is the dominant family, accounting for 1,605 samples (10.39%), followed by *timer* (808; 5.23%) and *revil* (650; 4.21%). Overall, the top 20 families comprise 10,291 samples, representing 66.62% of the entire corpus, while the remaining 33.38% is distributed across a long tail of less frequent families.

## VI. RANSOMWARE VS. GENERAL MALWARE CLASSIFICATION

To validate the distinctiveness of the **VERA** dataset, we designed a binary classification experiment to distinguish our verified ransomware samples from general malware. This establishes a baseline for detector performance and confirms that ransomware possesses unique structural and behavioral signatures distinguishable from generic malicious threats.

### A. Experimental Setup

*1) Dataset Balancing:* We constructed a balanced dataset by combining samples from two sources:

- **True Ransomware: 15,448** samples from the **VERA-Active** repository.
- **General Malware: 15,448** samples randomly sampled from the **BODMAS** dataset [5], which contains diverse malware families (Trojans, Worms, etc.).

This resulted in a total dataset of **30,896** samples, ensuring that model performance is not skewed by class imbalance.

*2) Feature Extraction:* We utilized the `pefile` library to extract a comprehensive set of **548 static features** from the raw binaries. These features capture the structural characteristics of the Windows PE format, including:

- **PE Headers:** Characteristics from `FILE_HEADER` and `OPTIONAL_HEADER` (e.g., `Machine`, `TimeDateStamp`, `ImageBase`).
- **Section Analysis:** Entropy, raw size, and virtual size for up to 33 individual sections (e.g., `.text`, `.rdata`), allowing the model to detect packing anomalies.
- **Import/Export Tables:** Counts of imported DLLs and total functions, often indicative of malicious capability (e.g., cryptographic or network APIs).
- **Byte Histograms:** Global file entropy and byte frequency distributions to identify encryption or compression.

Missing values were imputed with zeros, and features were standardized using a standard scalar prior to training.

### *B. Model Selection and Training*

We evaluated four standard machine learning classifiers to benchmark performance:

- **Random Forest (RF):** An ensemble of 800 decision trees.
- **Support Vector Machine (SVM):** Using an RBF kernel ($C = 10, \gamma = 0.01$).
- **Multi-Layer Perceptron (MLP):** A neural network with three hidden layers (512, 256, 128 units).
- **XGBoost:** A gradient-boosted decision tree model optimized for speed and performance.

The dataset was split into an 80% training set (24,521 samples) and a 20% testing set (6,131 samples).

### *C. Baseline Results*

The results of the binary classification task are summarized in Table V. All models achieved exceptionally high performance, indicating that the structural signatures of ransomware in **VERA** are highly distinct from general malware.

TABLE V
BASELINE PERFORMANCE: VERA RANSOMWARE VS. BODMAS GENERAL MALWARE

| Model | Accuracy | Precision | Recall | ROC-AUC |
|---|---|---|---|---|
| SVM (RBF) | 98.09% | 99.80% | 96.38% | 0.9996 |
| MLP | 99.80% | 99.90% | 99.71% | 0.9993 |
| Random Forest | 99.99% | 99.99% | 99.99% | 0.999 |
| **XGBoost** | **100.00%** | **100.00%** | **100.00%** | **1.0000** |

Both **Random Forest** and **XGBoost** achieved near-perfect classification scores on the test set. This suggests that without obfuscation, modern ransomware binaries possess significantly different static properties—such as specific section entropies and import table compositions—compared to general malware. The **SVM** model, while slightly less accurate (98.09%), still demonstrated high precision, with the primary errors stemming from a slightly lower recall (96.38%).

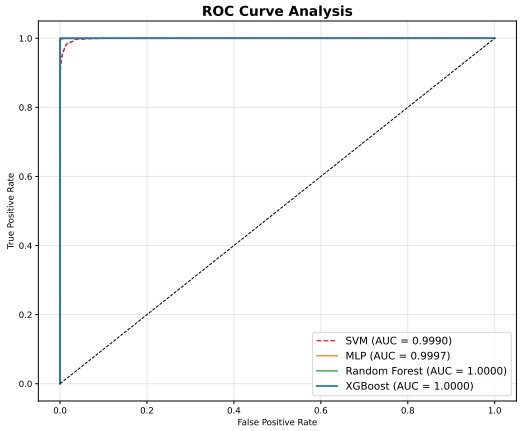


Fig. 5. ROC Curves for Ransomware Classification. The curves for Random Forest and XGBoost show ideal separation (AUC=1.0), while SVM and MLP show near-perfect performance.

While several models achieve near-perfect performance, we emphasize that this experiment is intended as a *sanity check* for label purity rather than a real-world deployment benchmark. Perfect separability suggests that static PE features may capture dataset-specific artifacts or distributional differences rather than robust ransomware semantics. We therefore treat this evaluation as a dataset validation signal rather than a deployment claim, and we plan robustness testing (packing/obfuscation/cross-dataset splits) as future work.

These results confirm that the **VERA** dataset contains high-fidelity samples with strong, learnable signals. However, such perfect baseline scores also highlight the potential fragility of static features, motivating the need for the evasion resilience testing discussed in future work.

## VII. FUTURE WORKS

While VERA employs a fixed three-minute execution window to balance coverage and scalability, longer or multi-phase execution strategies may further reduce false latency effects. In particular, ransomware samples that rely on delayed triggers, extended sleep loops, or environment-dependent execution conditions may be classified as *VERA-Latent* despite exhibiting malicious behavior under prolonged or more realistic deployment scenarios. Similarly, the presence of anti-sandbox and anti-analysis techniques may suppress observable behavior within constrained execution environments, artificially inflating the Latent category.

Future work will explore adaptive execution strategies, such as staged or event-driven sandboxing, diversified environment profiles, and temporal behavioral aggregation, to more accurately capture delayed or evasive ransomware behavior. Integrating these techniques would further refine the Active–Latent distinction and support more robust evaluation of ransomware viability under realistic adversarial conditions.

## VIII. Conclusion

In this paper, we presented **VERA**, a large-scale and artifact-rich repository of Windows ransomware binaries paired with dynamic execution logs. VERA is constructed using a dual-validation pipeline that combines static multi-vendor consensus from VirusTotal with automated behavioral verification in a CAPE Sandbox environment. A key outcome of this process is an explicit separation between **VERA-Active** samples, which satisfies both the static and bevahioural criteria, and **VERA-Latent** samples, which remain behaviorally silent within the analysis window. This distinction enables controlled studies of behavioral viability, delayed-trigger behavior, and the practical limitations of static labeling.

To support reliable family-level research, we further introduced a dual-stage family labeling pipeline that first builds a high-confidence consensus set using agreement across multiple label unification tools and then assigns labels to contested set via supervised learning over combined static and behavioral features. Along with raw binaries, execution reports, derived features, and metadata, we release the dataset and accompanying scripts to facilitate reproducible experimentation.

Finally, we evaluated the separability of verified ransomware from general malware using standard static PE features and common classifiers. The resulting near-perfect performance serves as a sanity check for label purity and supports the distinctiveness of **VERA-Active**, while also motivating future evaluations under obfuscation, packing, and cross-dataset generalization. Going forward, we plan to expand **VERA** with additional families and broader execution conditions, and to use it as a foundation for developing ransomware classifiers that remain robust under realistic evasion strategies.

## References


[1] Federal Bureau of Investigation, "Internet crime report 2023," Internet Crime Complaint Center (IC3), Tech. Rep., 2023. [Online]. Available: https://www.ic3.gov/annualreport/reports/2023_ic3report.pdf

[2] Y. Hou, L. Guo, C. Zhou, Y. Xu, Z. Yin, S. Li, C. Sun, and Y. Jiang, "An empirical study of data disruption by ransomware attacks," in *2024 IEEE/ACM 46th International Conference on Software Engineering (ICSE)*. IEEE, 2024, pp. 1–12.

[3] H. S. Anderson and P. Roth, "Ember: An open dataset for training static pe malware models," in *arXiv preprint arXiv:1804.04637*, 2018.

[4] R. J. Joyce, G. Miller, P. Roth, R. Zak, E. Zaresky-Williams, H. Anderson, E. Raff, and J. Holt, "Ember2024 - a benchmark dataset for holistic evaluation of malware classifiers," in *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining V.2*, ser. KDD '25. ACM, Aug. 2025, p. 5516–5526. [Online]. Available: http://dx.doi.org/10.1145/3711896.3737431

[5] L. Yang, A. Ciptadi, I. Laziuk, A. Ahmadzadeh, and G. Wang, "Bodmas: An open dataset for learning based temporal analysis of pe malware," in *4th Deep Learning and Security Workshop*.

[6] R. Harang and N. McLaughlin, "Sorel-20m: A large-scale benchmark dataset for malicious pe detection," in *Proceedings of the ACM Workshop on Artificial Intelligence and Security*. ACM, 2020, pp. 1–8.

[7] M. Sebastián, R. Rivera, P. Kotzias, and J. Caballero, "Avclass: A tool for massive malware labeling," in *Proceedings of the 19th International Symposium on Research in Attacks, Intrusions, and Defenses (RAID)*. Springer, 2016, pp. 230–253.

[8] S. Sebastián and J. Caballero, "Avclass2: Massive malware tag extraction from av labels," in *Annual Computer Security Applications Conference (ACSAC)*, pp. 42–53.

[9] M. Hurier, G. Suarez-Tangil, S. K. Dash, T. F. Bissyandé, Y. Le Traon, J. Klein, and L. Cavallaro, "Euphony: Harmonious unification of cacophonous anti-virus labels," in *Proceedings of the 12th International Workshop on Information Forensics and Security (WIFS)*. IEEE, 2017, pp. 1–6.

[10] FutureComputing4AI, "Claravy: A natural language processing approach to malware labeling," https://github.com/FutureComputing4AI/ClarAVy, 2024, accessed: 2026-01-20.

[11] 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.

[12] 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*, vol. 40, p. 301314, 2022.

[13] M. Hirano and R. Kobayashi, "Ransmap: Open dataset of ransomware storage and memory access patterns for creating deep learning based ransomware detectors," *Computers & Security*, vol. 150, p. 104202, 2025.

[14] J. Li *et al.*, "Mlran: A real-time behavior-based benchmark dataset for ml/dl-based ransomware detection," *arXiv preprint arXiv:2407.15949*, 2024.

[15] J. A. Herrera-Silva, "Dynamic feature dataset for ransomware detection using machine learning algorithms," *Sensors*, vol. 23, no. 3, p. 1053, 2023.

[16] C. J. W. Chew, V. Kumar, P. Patros, and R. Malik, "Real-time system call-based ransomware detection," *International Journal of Information Security*, vol. 23, no. 3, pp. 1839–1858, 2024.

[17] A. Alraizza and A. Algarni, "Ransomware detection using machine learning: A survey," *Big Data and Cognitive Computing*, vol. 7, no. 3, p. 143, 2023.

[18] S. Alzahrani, Y. Xiao, S. Asiri, N. Alasmari, and T. Li, "Ransomformer: A cross-modal transformer architecture for ransomware detection via the fusion of byte and api features," *Electronics*, vol. 14, no. 7, p. 1245, 2025.

[19] S. Gülmez *et al.*, "Xran: Explainable deep learning-based ransomware detection using dynamic analysis," *Computers & Security*, 2024, open access.

[20] M. Altaf, "Randel: Dynamic feature-based ransomware detection using ensemble learning," Preprints.org, 2025, preprint, Preprints.org manuscript 202502.1824.

[21] VX Underground, "VX Underground," https://vx-underground.org/, 2024.

[22] VirusShare, "VirusShare," https://virusshare.com/, 2024.

[23] MalwareBazaar. (2026) abuse.ch Project. abuse.ch / Spamhaus. Accessed: 2026-02-10. [Online]. Available: https://bazaar.abuse.ch/

[24] MalShare, "A free Malware repository providing researchers access to samples, malicious feeds, and YARA results." https://malshare.com/, 2019.

[25] VirusTotal, "Virustotal: A free online virus, malware and url scanner," https://www.virustotal.com, 2012, accessed: 2026-02-10.

[26] K. O'Reilly and A. Brukhovetskyy, "CAPE: Malware Configuration And Payload Extraction." [Online]. Available: https://github.com/kevoreilly/CAPEv2