Data in Brief 63 (2025) 112273

Contents lists available at ScienceDirect

Data in Brief

journal homepage: www.elsevier.com/locate/dib

Data Article

A dataset of windows malware execution traces

Razvan Raducu, Alain Villagrasa-Labrador, Ricardo J. Rodríguez, Pedro Álvarez*

Computer Science and Systems Engineering Department, Engineering Research Institute of Aragón (I3A), University of Zaragoza, Spain

ARTICLE INFO

Article history:
Received 27 October 2025
Revised 5 November 2025
Accepted 6 November 2025
Available online 13 November 2025

Dataset link: WinMET (Original data)

Keywords:
Malware dynamic analysis
Behavioral execution trace
System calls
Windows API

ABSTRACT

Malware continues to be a major cybersecurity concern, with increasing volume and sophistication making effective detection methods essential. Behavior-based approaches rely on high-quality execution trace data to analyze how malicious software interacts with systems during runtime. Publicly available datasets often lack sufficient detail, contain limited family diversity, or provide only simplified API call sequences. In this paper, we present a dataset that addresses this gap by offering a large collection of richly detailed Windows malware execution traces generated in controlled environments. It has been generated through automated dynamic analysis, executing the malware samples in a controlled virtualized environment, specifically, in the CAPEv2 Sandbox on Windows 10 virtual machines. The raw sandbox analysis reports have been then processed using the MALVADA framework, a modular Python-based pipeline that filters, structures, labels, and standardizes execution traces. The resulting dataset consists of 31,844 JSON execution trace files where each trace contains static metadata, dynamic behavioral information, and labelling fields. The dataset is suitable for reuse in multiple research contexts, including the development and benchmarking of malware detection methods, behavioral clustering, dynamic analysis of malicious software, and automated labelling studies. Its standardized JSON

* Corresponding author.
E-mail address: alvaper@unizar.es (P. Álvarez).
Social media: @Razvieu (R. Raducu), @Str1ien (A. Villagrasa-Labrador), @RicardoJRdez (R.J. Rodríguez)

https://doi.org/10.1016/j.dib.2025.112273

structure facilitates integration with existing data analysis and machine learning pipelines, as well as combination with other datasets for extended studies.

Specifications Table

Subject	Computer Sciences
Specific subject area	Cybersecurity, Malware analysis and recognition
Type of data	JSON format
Data collection	The dataset was generated executing malware samples in a virtualized controlled environment and monitoring the exhibited activity. Specifically, KVM was used to deploy several virtual machines (VMs) running Windows 10 × 64, and the CAPEv2 sandbox to automate sample submission, execution and activity report generation. Each of these reports contain key events occurred during the sample execution: created processes, sequence of API invocations, used system resources, network communications, etc. Subsequently, the MALVADA tool was used to curate, filter and label these malware reports. The labelling was based on the functionality of the AVClass tool.
Data source location	The analyzed malware samples were obtained from several publicly available malware repositories like VX-underground, Malware Bazaar, VirusShare, theZoo, and MalShare.
Data accessibility	Repository name: WinMET (Windows Malware Execution Traces) Dataset Data identification number: 10.5281/zenodo.12647555 Direct URL to data: https://doi.org/10.5281/zenodo.12647555
Related research article	Razvan Raducu, Alain Villagrasa-Labrador, Ricardo J. Rodríguez, Pedro Álvarez, *MALVADA: A framework for generating datasets of malware execution traces*, SoftwareX, Volume 30, 2025, 102,082, ISSN 2352–7110, https://doi.org/10.1016/j.softx.2025.102082. [1]

1. Value of the Data

- The Windows operating system is one of the main targets for malware [2], as it is the most used OS worldwide [3]. This dataset contains comprehensive behavioral information of approximately 32,000 execution traces of Windows malware samples. Each trace includes detailed contextual data such as API call sequences with arguments and return values, process trees representing spawned processes, accessed files and registry keys, synchronization objects, and network communications. This level of detail goes beyond simplified datasets that are available for the scientific community which typically only provide API names or identifiers.
- This dataset has been generated using a reproducible process which is based on an ecosystem of open source and publicly available tools. The process is structured as a modular pipeline with clearly defined steps, including malware execution, report generation, incorrect report detection, duplicate filtering, sensitive data anonymization, trace generation in JSON format, malware family labeling, and statistical reporting. The reproducibility of this pipeline facilitates the replication of the dataset creation process by other researchers, allowing the execution of new malware samples and the generation of additional execution traces to enrich and update the dataset.
- Traces of the dataset have been labelled with metadata describing their corresponding malware family. CAPE [4] and AVClass [5] have been used in the labelling procedure. These labelled traces are particularly suitable for malware recognition and detection based on Artificial Intelligence (AI) techniques. The dataset's scale (approx. 32,000 samples) and diversity of families and behaviors support a variety of data-driven research tasks, including feature

Table 1
Organization of the dataset.

File / Volume	Type / Format	Contents / Purpose
WinMET_volume_1.7z ... WinMET_volume_5.7z	7-zip archives (password protected: "infected")	Each volume contains a subset of the execution trace JSON files (total across volumes: 31,844 JSON traces).
avclass_report_to_label_mapping.json	JSON	Mapping of each execution trace to the label assigned by AVClass.
cape_report_to_label_mapping.json	JSON	Mapping of each execution trace to the label assigned by the CAPE sandbox labeling algorithm.
reports_consensus_label.json	JSON	For each execution trace, provides both its CAPE and AVClass labels.

extraction, supervised and unsupervised learning, and benchmarking of classification models. Additionally, JSON formatting simplifies preprocessing for sequence modeling, graph-based learning, and statistical analysis.

- The dataset can be used by researchers in cybersecurity, data science, and software engineering. Its detailed traces allow dynamic malware analysis, API sequence mining, labeling standardization, and visualization of malware behavior. The standardized JSON format facilitates integration into analytical pipelines, visualization tools, and machine learning workflows without requiring complex conversions. The dataset includes both behavioral data and metadata, enabling its use in multiple research contexts such as behavioral clustering, anomaly detection, attack reconstruction, and tool development. The open and structured nature of the dataset supports collaboration between different research areas and institutions.

2. Background

The growing sophistication and frequency of cyberattacks have intensified the need for advanced mechanisms able to prevent intrusions and accurately detect the type of malware responsible for them. The development and evaluation of such mechanisms strongly depend on the availability of high-quality datasets. However, existing malware execution datasets are often limited in size and the type of contextual information that include. Consequently, there is a clear need for new datasets that capture detailed behavioral traces of malware and are openly available to the research community. Furthermore, the maintainability and extensibility of these datasets should be guaranteed over time to support the continuous research progress and more robust and reliable detection strategies capable of effectively responding malware evolution.

The theoretical background of this work is grounded in behavior-based malware analysis, which examines sequences of API calls and system interactions to characterize and differentiate malicious programs. Methodologically, malware samples are executed in a controlled sandbox environment to monitor and capture their execution behavior. This behavior is subsequently structured in traces that include metadata related to the characterization and typology of malware. These resulting traces are the starting point to understand, interpret and analyze malware execution behavior.

3. Data Description

The dataset is hosted on Zenodo [6], an open-access and multi-disciplinary online repository. It comprises execution trace data in JSON format and files containing malware family labels. The traces are compressed into several archive volumes. Table 1 describes the organization of the dataset and the supplemental files.

Table 2
Most important JSON keys of an execution trace.

JSON Key	Description
detections	Array containing antivirus detection results and related metadata, derived from sandbox analysis and VirusTotal.
target	Object containing metadata about the analyzed malware file, including cryptographic hashes, PE structure, imported functions, strings, and VirusTotal scan results.
target.file.md5 / sha256 / ssdeep	Cryptographic hashes (MD5, SHA-256, SSDEEP) used to uniquely identify the malware sample and detect duplicates.
target.file.imports	List of imported libraries (DLLs) and their functions, including addresses and names, extracted from the PE header.
target.file.pe	Information about the Portable Executable (PE) structure, including resources and sections.
target.file.strings	Strings extracted from the binary, which can provide indicators of functionality or obfuscation.
target.file.virustotal	Object containing VirusTotal scan information, including scan ID, number of positives, total engines used, and vendor signature results.
dropped	Array of files or artifacts dropped or created by the malware during execution.
behavior	Object containing detailed dynamic analysis data gathered during execution, including processes, API calls, process trees, and summaries of accessed resources.
behavior.processes	Array of processes spawned during execution. Each process entry includes a unique ID, parent ID, and a list of API calls invoked.
behavior.processes.process_id	Numerical identifier of the process analyzed within the trace.
behavior.processes.calls	List of individual API calls made by a process during execution, including categories, names, arguments, and return values.
behavior.processes.calls.category	Broad category of the API call (e.g., filesystem, network, registry, synchronization).
behavior.processes.calls.api	Name of the specific API function invoked.
behavior.processes.calls.arguments	Array of argument name–value pairs passed to the API function during execution.
behavior.processes.calls.return	Return value from the API call, captured at runtime.
behavior.processtree	Hierarchical structure showing parent–child relationships among processes spawned during execution.
behavior.summary	Aggregated summary of system resources accessed by the sample, including files, registry keys, mutexes, executed commands, and other artifacts.
avclass_detection	Malware family label assigned by the AVClass tool, derived from VirusTotal vendor signatures.
Additional fields (...)	Depending on the sandbox configuration, additional metadata may be included, such as network activity, synchronization primitives, or extended analysis artifacts.

The traces are split into 5 vol, each containing approximately 6369 JSON files (except the final one with 6368). Compressed size per volume is about 2.5 GB, uncompressed size per volume is about 154 GB. Total compressed size: ≈13 GB; total uncompressed size: ≈750 GB for all volumes combined. Each JSON file in the volumes is an execution trace. A trace contains (among other keys) metadata about the executed sample (identification hashes, imported DLL or information about the executable structure), the dynamic execution behavior (processes, sequence of API calls, parameters and results of those API invocations, accessed resources, etc.), and labeling information (results returned by vendors and the malware family labels determined from those results). Table 2 lists the most relevant JSON entries for each execution trace.

The dataset includes a large and diverse set of malware families, as identified by both the AVClass and CAPE labeling algorithms. Fig. 1 and Fig. 2 show the top 15 malware family labels assigned by AVClass and CAPE, respectively, ranked by the number of samples associated with each family. The percentages shown in the figures are calculated relative to the total number of labeled samples, excluding those assigned the "(n/a)" label, which is assigned samples that could not be identified as any malware family by some of the algorithms. These unlabeled samples are part of the dataset but are not represented in the figures.

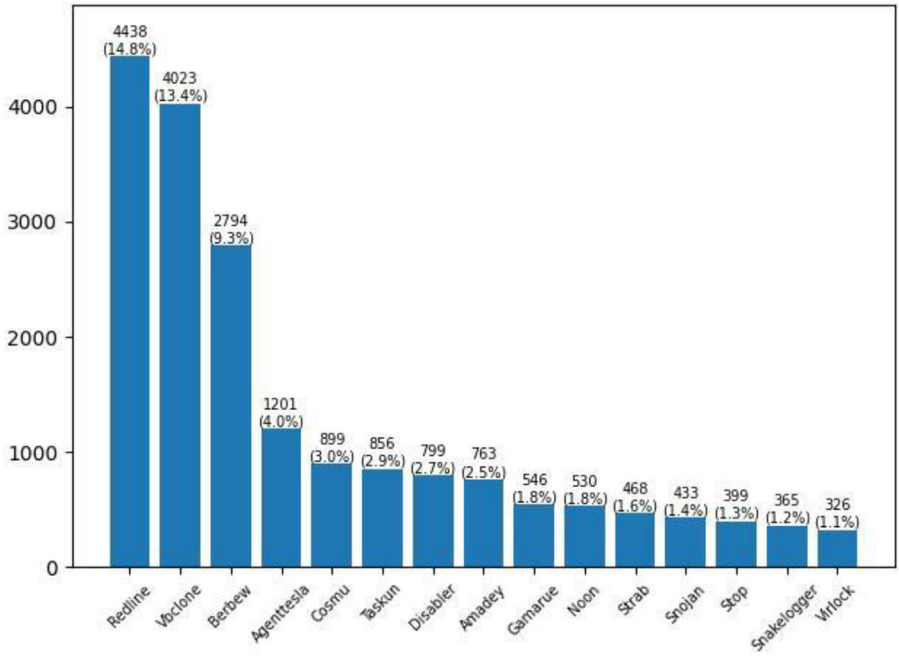

Fig. 1. Distribution of AVClass labels (percent w.r.t. total amount of execution traces).

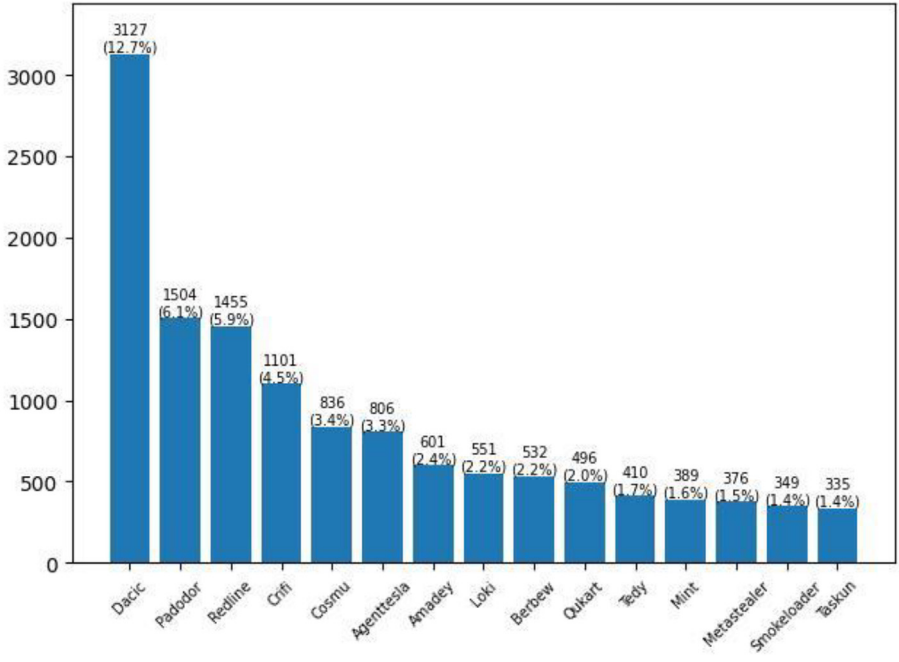

Fig. 2. Distribution of CAPE labels (percent w.r.t. total amount of execution traces).

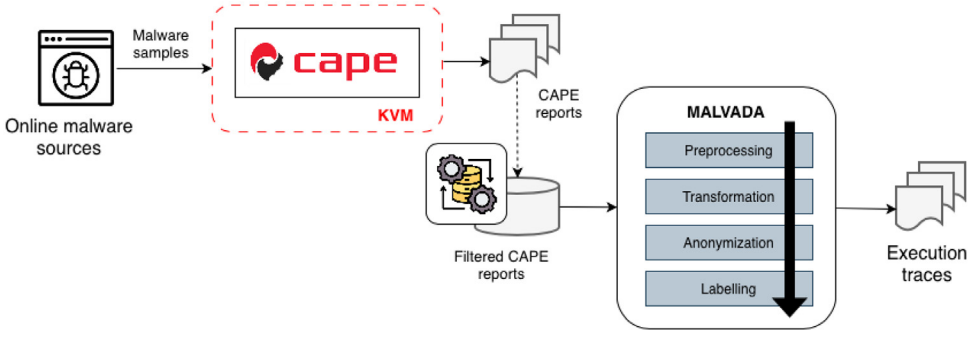

Fig. 3. Data acquisition process.

A comparison of both figures shows discrepancies in the families recognized by both labelling algorithms. AVClass infers malware families from the consensus of multiple antivirus engines, while CAPE identifies them based on dynamic behavior analysis. Therefore, these approaches apply different methodologies that often lead to variation in labelling. Nevertheless, these variations highlight the value of combining both perspectives, integrating a static (based on antivirus) and dynamic (on execution behavior) labelling. The result is reliable characterization of malware samples.

4. Experimental Design, Materials and Methods

The dataset was acquired by executing Windows malware samples in a controlled virtualized environment and collecting their execution traces. Those samples were obtained from several publicly available malware repositories such as *VX-underground, Malware Bazaar, VirusShare, theZoo*, and *MalShare*.

The process of execution trace generation consisted of three main stages that are represented in Fig. 3: (1) sandbox execution of malware samples using an enhanced version of the CAPEv2 Sandbox platform, (2) collection and preprocessing of dynamic analysis reports generated by CAPE, and (3) automatic processing and labeling of reports using the MALVADA framework to produce standardized JSON execution traces and associated family labels.

4.1. Environment for the execution of malware samples

A controlled malware analysis environment was deployed on a dedicated Ubuntu 22.04 LTS host using KVM virtualization. Samples were executed within Windows 10 × 64 virtual machines (VMs). Although Windows 10 was used at data-collection time, the setup is fully portable to Windows 11 VMs, facilitating future dataset updates and methodological continuity. While minor differences in system internals and API implementations exist between Windows 10 and Windows 11, these do not substantially affect the structure of the collected execution traces, as the CAPEv2 monitoring hooks target stable Windows API layers. Each VM was instrumented with a modified version of CAPEv2 Sandbox to monitor and log system activity generated by malware during runtime. We used the CAPE Hook Generator [7] tool to generate C-language hook skeletons, which were then inserted into CAPE's monitoring module (capemon) and recompiled. These hooks targeted additional APIs related to file and registry operations, network communications, memory usage and process injection, synchronization objects (e.g., mutexes, semaphores). These API invocations are necessary for understanding the execution behavior of malware, specifically, those related to the use of system resources and shared objects. The vir-

tual machines executed samples in isolation to prevent network interference or contamination between analyses. Incorrect executions due to crashes or connectivity errors were discarded automatically.

The result of sample executions was a collection of CAPE reports. These reports were stored in an internal data server to be preprocessed. This preprocessing mainly consisted of applying a set of filters to guarantee that the samples were executed correctly. Execution suspected of exhibiting evasion behaviors were discarded from the repository of reports.

4.2. Trace generation with MALVADA

The MALVADA tool was used to process CAPE reports in order to extract key data for understanding the malware execution behavior. This tool generates a detailed execution trace for each report which includes: the process tree, the API call sequence of each process, the parameters and results of those calls, accessed operating system resources, network communications, and used synchronization objects, among others. Besides, two labels are assigned to each trace for determining the malware family of the corresponding sample, generated by the labelling algorithms of CAPE and AVClass, respectively.

Internally, MALVADA was designed as a modular pipeline able to process efficiently large-scale collections of execution reports. The stages of the pipeline provide functionality to filter incorrect or duplicated reports, to transform reports into execution traces (in JSON format) by extracting and structuring relevant data about the execution behavior and context, to anonymize sensitive information, and to generate the malware family labels. Besides, the tool generates different statistics about the processing of the reports and the composition of the final dataset of traces.

MALVADA was programmed in Python. It works with minimal user intervention and offers a variety of configuration parameters related to its deployment and the report processing (a detailed description of these parameters can be found in [1]).

Limitations

The dataset is limited to execution traces generated in a controlled sandbox environment using a specific configuration of CAPEv2 Sandbox on Windows 10 virtual machines. As a result, those behaviors that rely on a certain type of user interaction besides the default mouse movement, system-specific configurations, or evasion techniques targeting sandboxes may not be fully captured.

Although the dataset contains over 30,000 traces, it does not cover the complete spectrum of existing malware families and variants, and the distribution of samples across families is uneven. Some families are represented by many samples, while others are less frequent.

The labelling of samples depends on CAPE and AVClass algorithms, which are based on vendors integrated into VirusTotal. These labels may contain inconsistencies or inaccuracies due to differences in vendor naming conventions and detection capabilities.

Finally, the dataset focuses exclusively on Windows malware and does not include benign software traces or malware targeting other platforms, which limits its representativeness for cross-platform behavioral studies.

Ethics Statement

The authors have read and follow the ethical requirements for publication in Data in Brief and confirming that the current work does not involve human subjects, animal experiments, or any data collected from social media platforms.

CRediT author statement

Razvan Raducu: Conceptualization, Data curation, Investigation, Methodology, Resources, Software, Supervision, Validation, Visualization, Writing – original draft, Writing – review & editing; **Alain Villagrasa-Labrador**: Data curation, Investigation, Resources, Software, Validation, Writing – review & editing; **Ricardo J Rodríguez**: Funding acquisition, Investigation, Project administration, Writing – review & editing; **Pedro Álvarez**: Conceptualization, Data curation, Funding acquisition, Investigation, Project administration, Supervision, Validation, Visualization, Writing – review & editing.

Data Availability

WinMET (Original data) (Zenodo)

Acknowledgments

This work was supported in part by grant PID2023-151467OA-I00 (CRAPER), funded by MICIU/AEI/10.13039/501100011033 and by ERDF/EU, by TED2021–131115A-I00 (MIMFA), funded by MCIN/AEI/10.13039/501100011033, by the Recovery, Transformation and Resilience Plan funds, financed by the European Union (Next Generation), by the Spanish National Cybersecurity Institute (INCIBE) under "Proyecto Estratégico de Ciberseguridad – CIBERSEGURIDAD EINA UNIZAR", and under "Cátedra Internacional de Ciberseguridad UNIZAR", and by the University, Industry and Innovation Department of the Aragonese Government, Spain, under "Programa de Proyectos Estratégicos de Grupos de Investigación'" (DisCo research group, ref. T21–23R). The work of Razvan Raducu was supported by the Government of Aragon, Spain, through the Diputación General de Aragón (DGA) Predoctoral Grant, during 2021–2025.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

[1] R. Raducu, A. Villagrasa-Labrador, R.J. Rodríguez, P. Álvarez, MALVADA: a framework for generating datasets of malware execution traces, SoftwareX 30 (2025).
[2] AV-ATLAS, "Malware & PUA," 2025. [Online]. Available: https://portal.av-atlas.org/malware; [accessed October 7, 2025].
[3] GlobalStats, "Desktop operating system market share worldwide," 2025. [Online]. Available: https://gs.statcounter.com/os-market-share/desktop/worldwide; [accessed October 7, 2025].
[4] K. O'Reilly, "CAPE: malware configuration and payload extraction," [Online]. Available: https://github.com/kevoreilly/CAPEv2; [accessed October 7, 2025].
[5] S. Sebastián, J. Caballero, AVclass2: massive malware tag extraction from AV labels, Annual Computer Security Applications Confe, 2020. *rence (ACSAC)*, Austin, USA.
[6] R. Raducu, A. Villagrasa-Labrador, R.J. Rodríguez and P. Álvarez, "WinMET Dataset," 2025. [Online]. Available: https://doi.org/10.5281/zenodo.12647555; [accessed October 7, 2025].
[7] R. Raducu, R.J. Rodríguez and P. Álvarez, "CAPEv2 (capemon) hook(s) generator," 2024. [Online]. Available: https://github.com/reverseame/cape-hook-generator; [accessed October 7, 2025].